Cranfield University

Ian Alexander Bledowski

Frequency-division-multiplexing technique for imaging metrology

School of Engineering

PhD

2014

Supervisors: Prof. R. P. Tatam and Dr. S. W. James

Abstract

An algorithm to multiplex multiple image captures simultaneously onto a single image sensor at full frame resolution was developed for imaging metrology. Parseval's theorem was used to obtain the image intensity from image time-series of around typically 256 frames captured by the imaging sensor at typically 60 fps, though kHz frame rates are possible, hardware permitting. The time-series contained contributions from each image channel in the system, which were created by periodically modulating the intensity of the light source which defined that channel. The modulating time-series was converted to a frequency representation by Fourier transform and from that the channels could be identified by their peaks in the spectrum. Peaks corresponding to each channel were then isolated with a window function and Parseval's theorem applied on a pixel by pixel basis to convert the signal strength back to an image containing the information from that channel only.

The FDM algorithm was then applied to two imaging metrology methods. First, an in-plane, two-channel shearography system was multiplexed with FDM in such a way as to allow time-division multiplexed measurements to be taken on the same deformations with the same instrument so as to allow comparison of results from other methods. FDM was found to produce good quality results comparable with current methods. Interferometric planar Doppler velocimetry was performed, multiplexing the reference phase channel signal and a signal channel for both a wheel and a gas jet. FDM was found to suppress the effects of phase drifts in the system which would lead to velocity offsets in the results, and gave velocities which varied from the model by only up to ~5%.

Finally, an error analysis was performed on the FDM algorithm, comparing the technique with time-averaging and single image capture through simulation and practical methods. It was shown that FDM strongly suppresses the noise and background in a measurement, and can produce good images from low intensity signals.

It could be concluded that the FDM algorithm offers significant advantages over time-averaging a signal when applied to a multi-channel imaging metrology system.

There is only one way to find out…

Contents

Acknowledgements

I would like to thank my supervisors, Prof. Tatam and Dr. James, for giving me the opportunity to work with them towards my goal of attaining a PhD. Thanks should also be extended to Dr. Charrett and Dr. Francis who both also helped me immensely throughout my time in Cranfield. I am grateful too for the cooperation and assistance of everyone in the Department of Engineering Photonics who will all have at some time been pestered with my requests.

I would like to acknowledge the financial support provided by the EPSRC.

Publications

Journal articles

I. A. Bledowski, T. O. H. Charrett, D. Francis, S. W. James, and R. P. Tatam, "Frequency-division-multiplexing for multi-component shearography," Applied Optics, Vol. 52, Issue 3, pp. 350-358 (2013)

T. O. H. Charrett, I. A. Bledowski, S. W. James, and R. P. Tatam, "Frequency-division multiplexing for Interferometric Planar Doppler Velocimetry," Applied Optics, Vol. 53, Issue 20, pp. 4363-4374 (2014)

T. O. H. Charrett, I. A. Bledowski, S. W. James, and R. P. Tatam, "Frequency-division multiplexing (FDM) for interferometric referencing and multi-component measurements in Planar Doppler Velocimetry (PDV)," 17th International Symposium on Applications of Laser Techniques to Fluid Mechanics, Lisbon, Portugal (2014)

Conferences

I. A. Bledowski, T. O. Charrett, D. Francis, S. W. James, and R. P. Tatam, "Frequency division multiplexing in shearography," presented at the IOP meeting on Optical Metrology, Coventry, UK (2011).

I. A. Bledowski, T. O. Charrett, D. Francis, S. W. James, and R. P. Tatam, "Frequency division multiplexed shearography," poster at OPTIMESS 2012, Antwerp, Belgium (2012).

I. A. Bledowski, T. O. Charrett, D. Francis, S. W. James, and R. P. Tatam, "Frequency division multiplexing in interferometric planar Doppler velocimetry," presented at Photon 12, Durham, UK (2012).

Nomenclature

AU	Arbitrary units
C	The C programming language
CCD	Charge coupled device
CMOS	Complementary metal-oxide semiconductor
CW	Continuous wave
DC	Direct current (usually refers to the 0 Hz component of a DFT)
DEP	Department of Engineering Photonics, Cranfield University
DFT	Discrete Fourier transform
DGV	Doppler global velocimetry (alternative form to PDV)
DPSS	Diode-pumped, solid-state
DPV	Doppler picture velocimetry
DSPI	Digital speckle pattern interferometry (alternative form to ESPI)
ESPI	Electronic speckle pattern interferometry
FDM	Frequency-division multiplexing
FFT	Fast Fourier transform
FFTW3	Fastest Fourier transform in the West dynamic link library
FM	Frequency modulated
fps	Frames per second
FSR	Free spectral range
HeNe	Helium neon laser
I-PDV	Interferometric filter planar Doppler velocimetry
M-PDV	Molecular filter planar Doppler velocimetry

MZI	Mach-Zehnder interferometer
Nd:YAG	Neodymium doped yttrium aluminium garnet laser
OPD	Optical path difference
PDM	Polarisation-division multiplexing
PDV	Planar Doppler velocimetry
RMS	Root mean square
RPM	Revolutions per minute
TDM	Time-division multiplexing

1 Introduction

Imaging metrology is a broad field with a wide range of applications that can be considered to deliver multiple measurements of a quantity across a spatial distribution inside the field of view without mechanical contact to the object under test. Often, the advantage of these techniques lies in the production of full-field measurements where traditionally only point-like measurements have been made (for example, a Pitot tube measuring the velocity of air in a wind tunnel measures the velocity at the tip of the probe), providing a fuller description of the effects being studied than might otherwise have been obtained.

The characteristics of the optical instrument are selected through the configuration of the components. For example, it is often possible to modify an optical instrument to develop its capability from the measurement of a single component or small subset of components to fully three-dimensional measurements of a quantity. This is mostly achieved, though not exclusively, through taking additional measurements from different angles to the measurand. These can be made by either placing additional cameras (with the associated optics) around the measurand or through placing the light sources similarly, which is often the more economical option. Each view or illumination direction in a system is considered a channel, and obtaining data from numerous channels requires a method of reading from the channels in order to obtain information in a meaningful manner, and this is where multiplexing and demultiplexing are employed. Perhaps the most conceptually simple approach to this is time-division multiplexing, where each channel is read from in sequence, one after the other. This can be enough for many situations, though there are significant gains to be made by employing a more complex multiplexing scheme.

In this thesis, a novel method for multiplexing multiple interferometric measurement channels to a single camera is presented. Termed frequency division multiplexing (FDM), this method allows for images from multiple light sources to be captured simultaneously at the full available resolution of the camera, and the images produced from this method are suitable for applications in imaging metrology. The algorithm is first discussed from its development and initial testing, before being demonstrated in applications pertaining to stress analysis using shearography and to velocity measurement using planar Doppler velocimetry. Finally, the algorithm was analysed under various conditions to assess its performance and the results presented to give an indication of the errors a researcher might expect to encounter when utilising the technique.

Starting in chapter 2, the algorithm is developed from the principle of Parseval's theorem. This showed the relationship between the intensity of a signal before and after taking the Fourier transform of it, before demonstrating how this can be applied to determine the intensity distribution of a single image in an image time-series containing multiple images all with different modulation frequencies. A demonstration of the algorithm successfully separating two images and a measure of the noise improvements are also shown. In chapter 3, the FDM algorithm is applied to shearography. A short introduction and review is given to speckle interferometry as applied to shearography, and its applications, then discussion is made of transducing intensity to phase using temporal phase stepping algorithms. Phase unwrapping, fringe improvement and conversion of phase to strain is then discussed. Multiplexing techniques in shearography are reviewed before presentation of the experimental application of FDM shearography. In chapter 4, flow measurement is undertaken with FDM multiplexing applied to the technique of interferometric planar Doppler velocimetry (PDV). The field of PDV is reviewed briefly, before discussion is made of the measurement of phase using spatial techniques. Filtering methods for improving the quality of measurements follows after that, before the experimental apparatus is described. The results of the experimental applications of FDM to interferometric PDV are presented and discussed. Chapter 5 contains multiple error analyses of the FDM method. The effects of sampling in Fourier space are discussed, along with the effects of time averaging inherent in the Fourier transform process and how these affect image quality when reconstructing the information from that channel. Noise propagation through the algorithm is explored under various simulated conditions. The camera used in the experimental work is analysed, and the benefits of the FDM algorithm are discussed. Chapter 6 draws together the conclusions drawn throughout the thesis. Two appendices are attached containing the source code; appendix A contains that of the shared library used to speed the processing of the large data sets the FDM algorithm demands, and appendix B the source of the Python module used to enable FDM processing.

2 Algorithm

2.1 Introduction

The FDM algorithm was conceived to allow the multiplexing of multiple images to a single camera, utilising the full image resolution. This is achieved through detecting from an image time-series (similar to a video capture) the unique intensity modulation signal applied to each image being multiplexed to the camera. The technique relies upon Fourier analysis of the time-series at a given pixel in the image plane, selecting from the power spectrum calculated for that pixel each of the peaks corresponding to the applied modulations. The mathematical theory is described in this chapter and the approach is demonstrated experimentally by multiplexing images to establish the validity of the method.

While the algorithm was conceived of using principles of Fourier theory in order to multiplex images, there existed other Fourier based techniques that have achieved isolation of information from a measurement system. Fischer et al [1] measured, at multiple points in space, laser frequency modulations in the presence of a narrow molecular absorption feature to separate different components of velocity in Doppler global velocimetry. Takeda [2] used fringes in imaging through a path length imbalanced interferometer to reveal phase changes in a signal which were revealed through processing the spatial distribution of frequencies in Fourier space. The FDM algorithm presented here was, to the knowledge of the author, the first time images were multiplexed for imaging metrology in this manner.

2.2 Theory

FDM allows the simultaneous capture of data from multiple measurement channels of an imaging metrology instrument on a single camera sensor. The technique is applied in this body of work to two imaging metrology systems that utilise laser illumination of the object to be measured, which is imaged through an interferometer on to the camera sensor. Discrete intensity modulations in time are applied to the light before transmission to the measurand, which provides multiple measurement channels when the modulated light fields are transmitted simultaneously. The resulting images of the light scattered from the measurand are therefore received simultaneously at the camera sensor and their evolution over time is captured in a sequence of images recorded over a number of modulation cycles. The time-series of a given pixel, $I(t)$, will

vary in intensity over time as a function of the sum of all of the modulation frequencies present.

$$I(t) = \sum_{\mathrm{n}} I_n(t) + C \qquad \textit{2-1}$$

Where $I_n(t)$ is the modulation signal of channel n, and C is the background intensity.

To de-multiplex the channels, the power spectrum is calculated via the discrete Fourier transform (DFT) on a pixel-by-pixel basis. The DFT is defined as shown for the forwards and the backwards transforms respectively:

$$A_k = \sum_{m=0}^{N-1} a_m \exp\left(-\frac{2\pi m k}{N}\right), \qquad k = 0, \dots, N-1 \qquad \textit{2-2}$$

$$a_m = \frac{1}{N}\sum_{k=0}^{N-1} A_k \exp\left(\frac{2\pi m k}{N}\right), \qquad m = 0, \dots, N-1 \qquad \textit{2-3}$$

where a_m and A_k are the signal in real and Fourier space respectively, m is a sample in real space, k is the frequency sample in Fourier space, and N is the total number of sample points taken.

By applying the forward transform of equation 2-2 to equation 2-1, the frequency composition of the measured signal may be revealed for processing via the power spectrum. The peaks corresponding to the different channel modulation frequencies are separated using window functions and the intensity is evaluated. The root-mean-squared (RMS) intensity, I_{RMS}, of a channel's signal can be calculated in the frequency domain using Parseval's theorem (sometimes referred to as Rayleigh's theorem or the energy theorem), which equates the intensity of the measured signal (in this case, in time) with the intensity of the signal in the power spectrum as shown [3]:

$$\sum_{m=0}^{N-1} |a_m|^2 = \frac{1}{N}\sum_{k=0}^{N-1} |A_k|^2 \qquad \textit{2-4}$$

In the case of FDM, the signal that is measured in time contains contributions from all the channels, which is evident in the power spectrum generated, where peaks from each channel's modulation will be present. In order to obtain the signal from a single channel, an appropriate filter must be applied to select only the contribution of the desired channel. This function has been termed the *window function*, and is applied in the Fourier domain. The resulting relationship between the signal domain and the Fourier domain is shown:

$$I_{RMS} = \sigma_{I_n} = \sqrt{\frac{\sum_{\mathrm{m}} |I_n(t)|^2}{N}} = \sqrt{\frac{\sum_{\mathrm{k}} |G(f) \cdot H_n(f)|^2}{N^2}} \qquad 2\text{-}5$$

where $G(f) = \mathcal{F}\{I(t)\}$ is the DFT spectrum of the measured signal, $I_n(t)$, $H_n(f)$ is a window function to select the peaks of the modulation in the signal, and N is the number of samples. Multiplying the spectrum of the measured signal by the window function allows the desired signal to be isolated from any other information in the captured data. As the background light level at the zero frequency is common to all the channels, this DC component of the signal is removed when windowing in the frequency domain. The RMS signal calculated in equation 2-5 is thus equivalent to the standard deviation of the signal, σ_{I_n}, as it modulates in time. Hence, the peak-to-peak intensity, I_{pp}, of the modulating signal can be found by dividing the extracted RMS intensity by the RMS intensity of the normalised waveform, σ_{cal}, which can be used for calibration purposes.

$$I_{pp} = \frac{\sigma_{I_n}}{\sigma_{cal}} \qquad 2\text{-}6$$

The normalised waveform is not used in the practical measurements in the following chapters, rather is mentioned here to allow a means of comparison between peak-to-peak intensities.

In general, when applying the FDM algorithm, the image intensity is calculated for each channel present, pixel by pixel, and these values get placed into separate image arrays per channel. Due to the summing of the time-series and division by the number of samples, the demultiplexed images can then be considered equivalent to time-averaged images captured with only a single illumination channel present at any one time. The technique is summarised in Figure 2-1, which illustrates the steps of the demultiplexing process.

It should be noted that it is not always necessary to calculate the peak-to-peak amplitudes. Instead, the sum of the windowed power spectrum can be used to speed de-multiplexing by omitting steps from the RMS calculation, shown at step (d) part (ii) in Figure 2-1. However, the intensity, I_{NL}, obtained by this method is scaled non-linearly, which could be a problem when used with some phase stepping algorithms [4]. Another point to note is that the modulation frequencies should be chosen so that they meet the Nyquist condition and have resolvable peaks in the power spectrum.

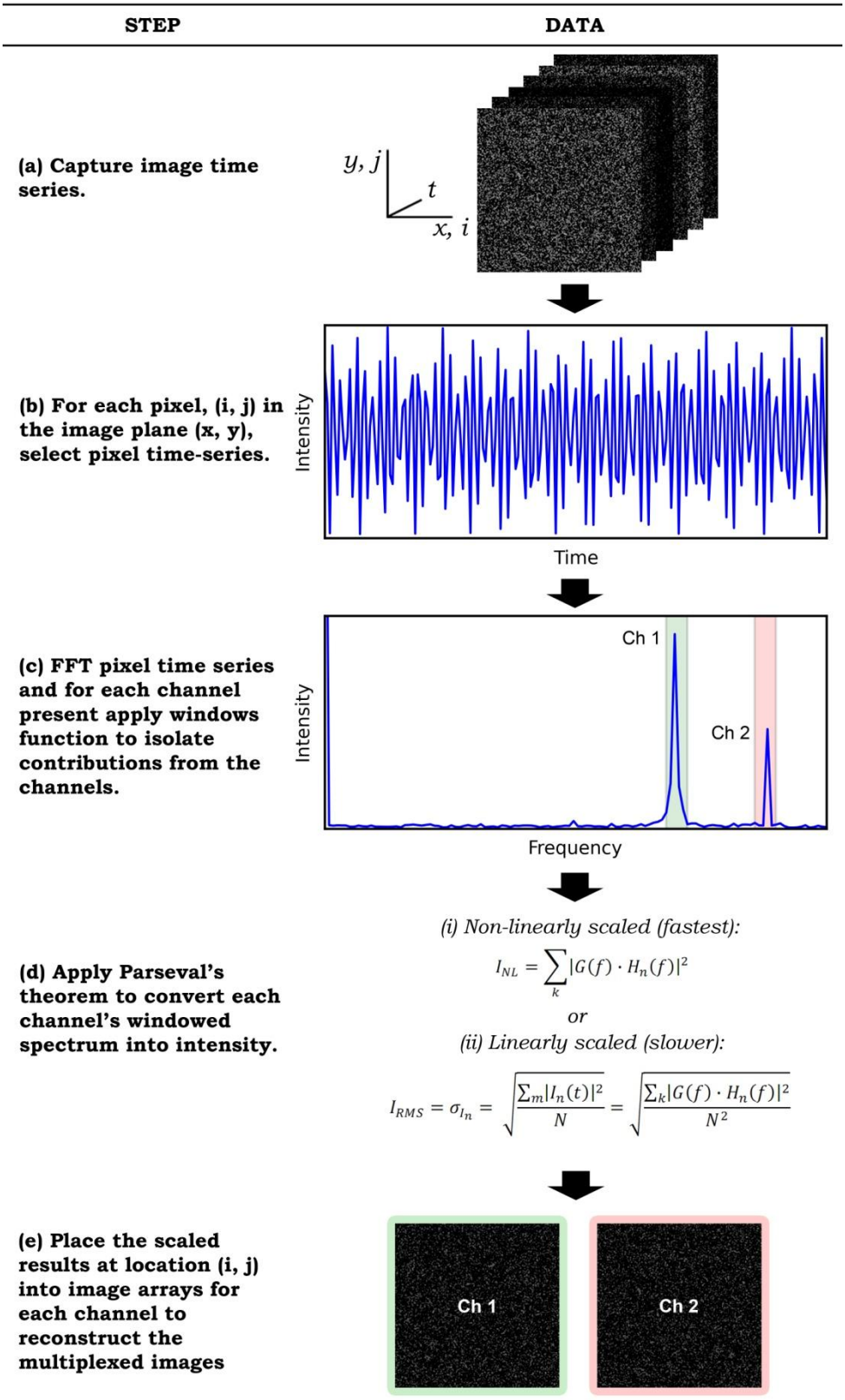


Figure 2-1. Steps to demultiplex images from an image time-series. The example shown is for demultiplexing two channels of speckle images.

2.3 Experimental implementation

The first step in demonstrating the viability of the multiplexing technique was to show that images could be reconstructed and separated from an image bank containing multiple images. To this end, an experiment to multiplex two images together on a single camera and subsequently process the information captured to reconstruct and demultiplex the images was designed. The results are presented in this section.

The experiment was configured as shown in Figure 2-2 using the expanded beams from two laser sources (a Coherent DPSS 532-300 with an output power of 200 mW at a wavelength of 532 nm and a Photop DPGL-3020 DPSS module with an output power of 20 mW at 532 nm) incident on a screen. The beam in each channel was transmitted through a spatial modulator to generate distinct static intensity distribution patterns. The patterns were produced from a calibration target slide's markings and from the coils of a spring. These patterns were imposed on each channel to act as a visual aid to confirm that the algorithm performed as expected. The patterned light fields were incident simultaneously on the screen and imaged on to a high speed camera (Baumer HXC-13) with a 144 x 150 pixel sub-region selected on the sensor allowing a frame rate of 11,000 fps. Images were recorded using a field of view of approximately 41 mm^2. The images in this experiment were both views of the same area of the screen, but under two different illumination conditions. This is a similar configuration to that found in many multiple component imaging metrology systems. Without applying the FDM algorithm, the two light fields in this experiment would be inseparable.

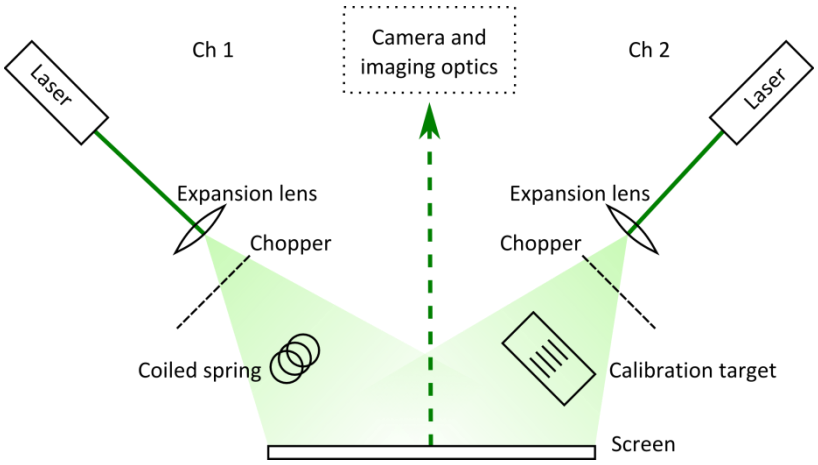


Figure 2-2: Experimental multiplexing of two images (shadows from a coiled spring and a calibration target projected on a screen).

The FDM algorithm was implemented with the application of time varying modulations on the respective beams. The outputs from the lasers were intensity modulated in time by placing a beam chopper in each beam path to produce square wave intensity modulations at rates of 1 kHz and 1.5 kHz in the respective channels. A long time-series of 1024 modulated images was captured, as processing time was not an issue at this stage. After capturing the time-series, individual images of the two light fields were captured as single-frames with only a single illumination channel present, allowing comparison of the demultiplexing approach with the conventional approach. The time-series images were post-processed with the computer implementation of the FDM algorithm to demultiplex the images of the two channels from the captured image time-series.

Development of the processing algorithm was conducted in the open-source Python programming language [5] using the Python Toolkit environment [6], with the NumPy and SciPy [7] modules. A bespoke shared library was then written in C using the FFTW3 library [8] in order to increase memory efficiency and the speed of the DFT based analysis. This library implemented the pixel-by-pixel application of the DFT stages of processing the time-series. This approach of combining languages was capable of de-multiplexing the channels on the order of seconds using a single core of a desktop PC running at 2.86 GHz. Typical data sets captured consist of large, three-dimensional arrays of signals with sizes that can approach one gigabyte during processing, requiring careful memory management.

The results from this can be seen in Figure 2-3, which shows the single-frame captured images (parts a and b) above their demultiplexed counterparts (parts c and d). It can be seen that the images are reconstructed with good fidelity. Also shown are

the de-multiplexed images when the less computationally intensive method of calculating intensity from the power spectrum is used (e and f), omitting the scaling process used in the calculation of the RMS intensity. These images show clearly that the non-linear method introduces a more rapid drop-off from high to low intensity in the demultiplexed images, as compared to the single-frame captures. If processing time is an issue (for example if attempting to process in real-time or there are time dependent sources of error (see an example of this in chapter 5)) then these images may be used with a temporal phase-stepping algorithm that is insensitive to non-linearities [4]. As the time gain was only a fraction of a second, the RMS scaled version of the calculation was adopted for use in post-processing.

Images can be demultiplexed using illumination channels originating from a common laser source, and also at a range of camera frame rates and modulation frequencies that meet the Nyquist condition.

Following this demonstration of the feasibility of the FDM approach, the FDM algorithm was applied to practical imaging metrology systems. The first system investigated was two-component in-plane shearography, reported in [9] and described in chapter 3. Noise analysis was performed the FDM technique following its use in making measurements. This practical noise performance of the system is given in this chapter as typical performance.

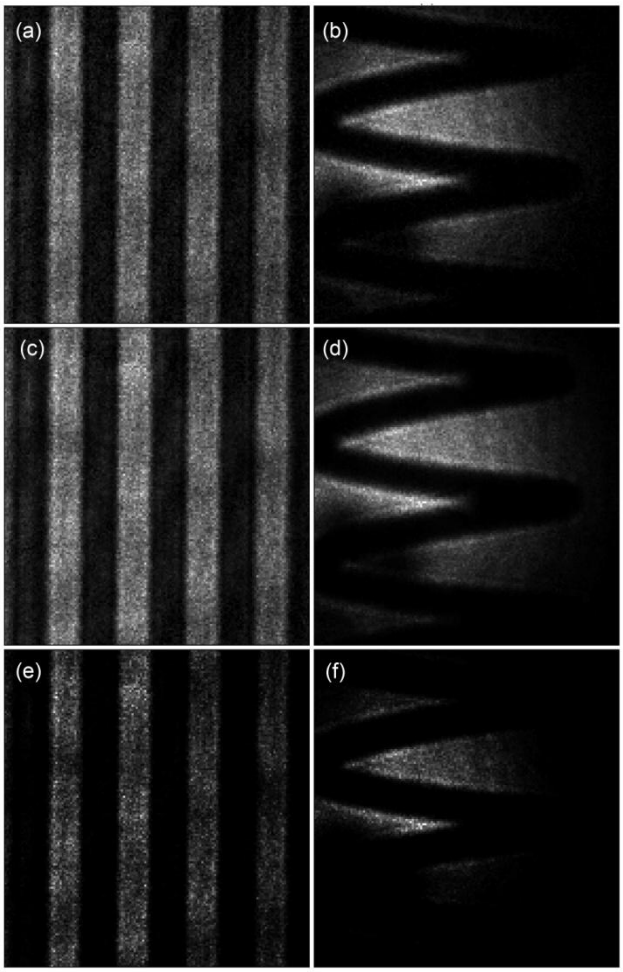


Figure 2-3. (a) and (b) are directly captured images of objects placed in the path of the sources illuminating the separate channels across a field of view of approximately 4 cm^2. (c) and (d) are de-multiplexed images of the same objects that had been multiplexed by chopping the beams at 1 kHz and 1.5 kHz respectively and that were captured simultaneously in an image time-series of 1024 frames at 11000 fps. The images in (e) and (f) show the same patterns de-multiplexed using the non-linear method, where intensity is the sum of the squares of the windowed power spectrum. Note the intensities produced by that method increase more sharply towards the bright regions due to the introduction of a non-linearity to the intensity calculation. Each image is scaled to the highest intensity in the frame.

2.4 Potential errors and uncertainties

As the de-multiplexing technique effectively filters the power spectrum when selecting the peaks, the camera noise spectrum will also be filtered. Due to this, the noise which remains in the de-multiplexed images is reduced compared with that of single frame

measurements. The camera used in this work (Baumer HXC-13) is a CMOS image sensor. Therefore each pixel has its own noise characteristics described by a mean, $\mu(i,j)$, and standard deviation, $\sigma(i,j)$, where i *and* j are the pixel array indices. To examine the noise characteristics practically, the camera was set to mimic the conditions used during the experimental demonstration of FDM shearography (described in chapter 3).

A lens cap was placed over the camera sensor and a time-series of 256 images was captured at a frame rate of 430 fps. From this dark time-series, two images were de-multiplexed using rectangular windows centred at 140 Hz and 175 Hz, both of width 10 Hz. The image time-series was also averaged along the time axis to produce an image of the average intensity at each pixel over the capture period. This time-averaged image was analysed with the aim of investigating whether the de-multiplexing method produced improved noise characteristics compared with calculation of the mean of a time-series. The level of the camera noise present in the de-multiplexed and the time-averaged images was compared to that of single-frame measurements, equivalent to TDM, one captured at the same time as each time-series. The whole procedure was repeated 100 times. This gave a set of de-multiplexed images from which the means, $\mu(i,j)$**,** and standard deviations, $\sigma(i,j)$, could be determined at each pixel over time. The values for the pixels were then averaged across the frame and the results presented in Table 2-1.

Overall, the de-multiplexing method was shown to reduce the mean noise floor of the camera compared to both the single-frame and the time-averaged images by a factor of fourteen. The consistency of the measurements, demonstrated by the average standard deviation of all the pixels, obtained by de-multiplexing was also greatly improved compared to that of the single-frame TDM approach, showing an improvement of a factor of twenty.

In comparison to time-averaging the same number of frames, the standard deviation of the demultiplexed data was less. However, while the FDM technique showed an improvement in the level of the noise floor, the signal to noise ratio was half that obtained by time averaging, due to the 50% duty cycle of the modulation signal with half the light used to multiplex the channels. While this may seem to place FDM at a disadvantage relative to the simpler time-averaging approach, it is worth noting that time-averaging is only possible on a single channel, whereas the FDM approach allows the capture of multiple images simultaneously in a single time-series of the same length. For the same total capture period, T, FDM can capture n channels with noise averaging over the whole image time-series capture period (equal to T) while time-

averaging would only be able to capture in the space of T/n per channel, greatly reducing the benefits of averaging. This could be of concern when measuring a dynamically evolving system.

Table 2-1: Noise characteristics of an average pixel calculated over 100 measurements for single-frame, time-averaged, and de-multiplexed techniques. TPB is the typical pixel background level, calculated as the mean background of all pixels i,j. TPSD is the typical standard deviation of an image pixel between measurements, calculated as the mean standard deviation.

	Single-frame	Time-averaged (256 frames)	De-multiplexed (f = 140 Hz)	De-multiplexed (f = 175 Hz)
TPB, $\langle \mu(i,j) \rangle$ (counts)	0.43	0.43	0.03	0.03
TPSD, $\langle \sigma(i,j) \rangle$ (counts)	0.134	0.010	0.007	0.007

A further source of error is the cross-talk between measurement channels. This was measured again during the shearography experiment, but can be considered for the purposes of this section as a situation in which the multiplexed speckle patterns were produced by illuminating a surface with two expanded beams split from a common laser source (Coherent DPSS 532 – 300), as shown in Figure 2-4.

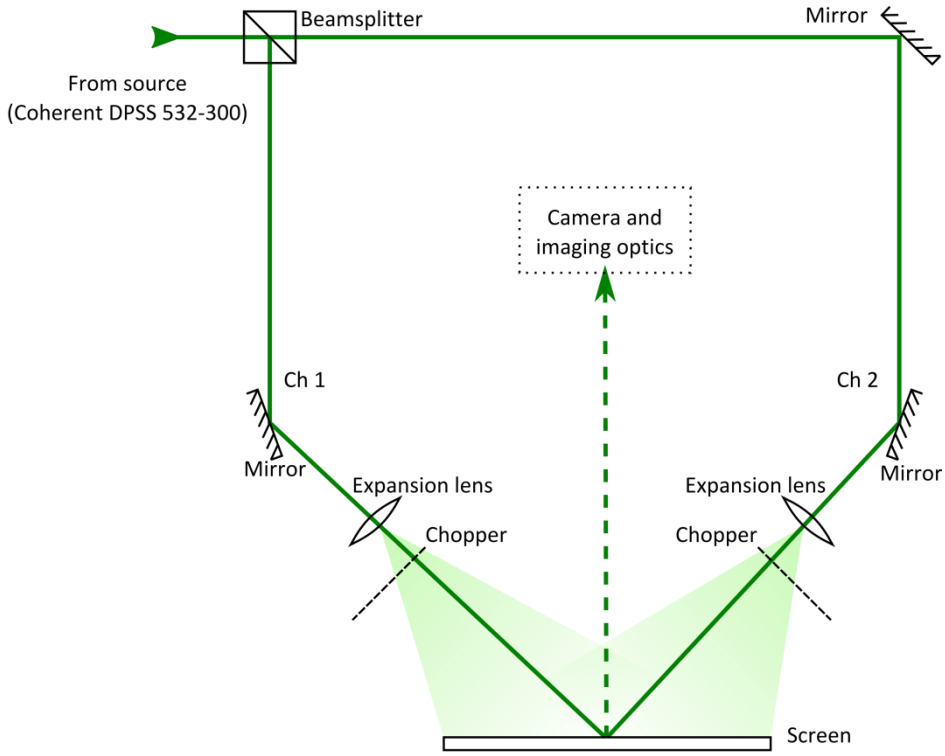


Figure 2-4: The experimental layout for measuring cross-talk between channels.

The intensities of the beams were modulated by placing mechanical choppers in the beam paths, resulting in modulation frequencies of 140 Hz and 175 Hz. As before, the camera (Baumer HXC-13) was set to capture image time-series consisting of 256 frames at a frame rate of 430 fps. The surface was illuminated in turn by each of the modulated laser sources, and a speckle pattern time-series was captured for each. From each time-series two speckle patterns were de-multiplexed. The first was extracted at the illuminating channel's modulation frequency and was used as a measure of the signal for that channel (I_{source}). The second speckle pattern was extracted at the frequency corresponding to the modulation frequency used for the second illuminating channel, were it present. This contained information consisting of cross-talk from the illuminating channel and contributions from the camera noise (I_{cross}). The ratio of the mean intensities in I_{cross} to I_{source} can be used as a measure of the cross-talk. Overall, cross-talk was measured to be an order of magnitude higher than the camera noise floor, with the cross-talk between channels, defined as the ratio of power leaked over the power remaining in the originating channel, measured to be 1.5% and 2.0% from channels one and two, respectively. The difference between these two values is most likely due to a localised increase in the spectrum intensity in one channel at the frequency the other channel was situated upon, increasing the intensity derived from application of the window there. This is usually caused by the presence of harmonic peaks in the DFT. A more in-depth discussion of cross-talk is

given in chapter 5. These crosstalk values compare favourably with polarisation-division multiplexing schemes, where a cross-talk of 20% has been reported from the light experiencing depolarisation on scattering from the surface [10]. Possible causes of cross-talk are mixing of higher order harmonics or overlap of peaks from different images.

2.5 Conclusions

It has been demonstrated in this chapter that an FDM algorithm based on Parseval's theorem can be applied successfully to demultiplexing images from an image time-series.

It has been shown to have an inherent noise suppression effect due to the time averaging and windowing of the signal DFT. The approach offers a significant advantage over simple time averaging of a single a channel, as multiple channels may be demultiplexed simultaneously and each channel experiences noise suppression from the entire number of frames captured in the experiment.

Levels of cross-talk were demonstrated to be lower than in polarisation division multiplexing, with around 2% of power leaking from one channel to another.

3 Application of frequency division multiplexing to shearography

The practical work in this chapter has been published as:

Ian A. Bledowski, Thomas O. H. Charrett, Daniel Francis, Stephen W. James, and Ralph P. Tatam, "Frequency division multiplexing for multi-component shearography," Applied Optics, vol. 53, no. 3, pp. 350-358.

3.1 Introduction

Shearography [11] is a non-contact full-field optical interferometric technique that allows the measurement of the gradients of displacement on surfaces subjected to loading. This is a property which lends itself well to the measurement of surface strain across an object in applications ranging from large field-of-view examinations of vibrational modes across aircraft panels [12], to quantitative measurements of strain on microscopic MEMS devices [13].

A speckle interferogram is obtained by illumination of an object's surface with a laser. As long as the surface is uneven on the scale of the wavelength of light, a speckle pattern is produced. The speckle pattern is imaged using an image shearing device, which is discussed further below, resulting in an interferometric speckle pattern that is sensitive to the surface displacement gradient. Processing the phase changes induced in interferograms taken before and after loading the surface produces a result that can be related to surface strain, and, by using multiple measurement channels, the components of surface strain can be measured separately and quantitatively.

In this chapter, a novel configuration for multiplexing the measurement channels of a multi-component shearography system, exploiting frequency division multiplexing (FDM), is described. This allows the simultaneous capture of speckle patterns from each measurement channel on a single imaging sensor without loss of spatial resolution. The theory of quantitative shearography is summarised and the application of FDM to shearography is described. Finally the results of experimentally applying FDM to a two-component shearography system are presented and discussed.

3.2 Speckle interferometry theory

3.2.1 Laser speckle

Speckle is a property of light observable when illuminating optically rough surfaces, and results in a granular appearance to the intensity distribution of the diffuse reflection. It is most commonly observed in laser light, but can also be observed in white light. By Huygens principle, an illuminated surface can be thought to reflect the incident light at each illuminated point as spherical secondary wavefronts that combine to produce a random interference pattern. Speckle patterns can be described as either subjective or objective depending respectively on whether the light is imaged or not (see Figure 3-1) [14][15].

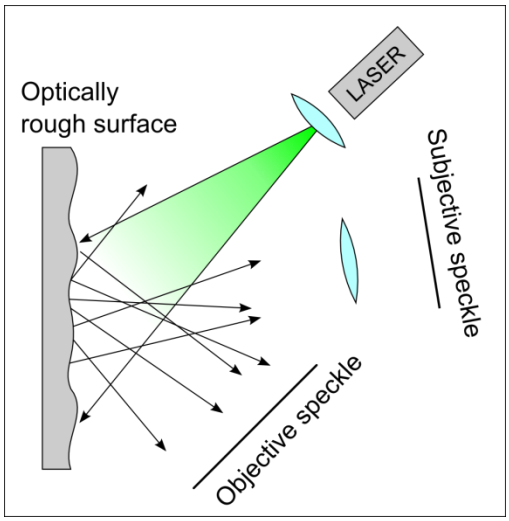


Figure 3-1: Subjective and objective speckle pattern origins. Arrows signify the normal to the wavefronts.

The phase distribution ϕ across an objective speckle pattern can be described by the following relationship to the optical path difference (OPD) of the light with wavelength λ [15]:

$$\phi = 2\pi k \frac{OPD}{\lambda} \qquad 3\text{-}1$$

Where the surface roughness determines the OPD, and k is known as the sensitivity vector. Hence a low roughness surface will not produce a speckle pattern. Phase is constrained within an interval of $-\pi$ to π. The average diameter of the observed speckles is dictated by the diameter of the illuminated area D and the distance L from the surface, and can be approximated by $\sigma \approx L\lambda/D$. A subjective speckle pattern is, by contrast, dependant on the imaging optics used, as a smaller region of the reflected

light is sampled by the lens due to the focal length f and the aperture D_a, in which case the speckle diameter is approximated by $\sigma \approx f\lambda/D_a$. The resulting pattern is therefore dependent on the spatial frequency components allowed to pass through the aperture and on the diameter of the Airy disc. Control of speckle size can be achieved in subjective speckle patterns by varying the aperture diameter, allowing it to be matched to the pixel dimensions of a CCD camera in order to optimise sampling across the image.

Speckle patterns are inherently noisy, with phase noise being random with a uniform distribution across the range $-\pi$ to π, with the result that the signal to noise ratio of the pattern is unity [14]. Phase information is encoded in the random structure of speckle patterns, which allows them to be used interferometrically, though speckle can be a source of noise in some forms of measurement.

The appearance of a speckle pattern can be changed by the properties of the surface that scatters the light [16]. For example, increasing the surface roughness has the effect of introducing multiple scattering which depolarises the scattered light, making the speckle pattern appear different when viewed at two polarisation states. Speckle patterns analysed for two states of polarisation are increasingly decorrelated for increasing depolarisation, though for scattering off less rough surfaces the speckle patterns remain correlated. As depolarisation increases, the contrast of the speckle pattern decreases. This is due to the light in the unpolarised component of the light field summing to a uniform intensity without speckle.

In order to extract the useful information, techniques that reduce the effects of noise on the speckle pattern must be employed. When imaging the pattern, noise can be reduced by taking and then averaging multiple exposures. Various noise reduction techniques can also be employed in the processing of speckle images. Methods based around phase fringe improvement by convolution with a smoothing kernel are discussed in section 3.4.3.

3.2.2 Early work in speckle techniques

An early, influential paper in the area of the use of wavefronts to determine surface characteristics was penned in 1974 by Bruning et al [17], which described a computerised method for scanning the speckled wavefronts generated from a lens under testing by a Twyman-Green interferometer and determining the relative phase changes across the wavefront to an accuracy of $\lambda/100$.

Early systems would record holographic data on photographic plates, which would require double exposure to encode the data and Fourier optics techniques to extract the fringes for a qualitative analysis. Later systems dispensed with photographic plates to avoid chemical processing, using instead thermoplastic plates that acted similarly to a photograph but were reusable. Laser speckle was an undesired property of the illumination, causing a reduction in quality of the holographic images.

Use of these holographic techniques declined with the advent of silicon imaging arrays which allowed computer based processing and led eventually to quantitative analysis of the images. Speckle was no longer a hindrance to measurements made using silicon chips, as it was possible to extract interferometric information from speckle and display results faster than could be achieved with holography. Thus electronic speckle pattern interferometry came to prominence [14].

3.2.3 ESPI

Born out of the research of Bruning, the field of wavefront interferometry developed into holographic interferometry, which involved the capture of light fields reflected from a test object that were mixed with a reference beam (typically a beam with a uniform phase distribution sampled from the illumination beam before transmission) on holographic and photographic plates. Object displacement–induced phase changes in the speckle pattern relative to the reference beam caused intensity variations in the speckles, which could then be captured in either three dimensions on holographic plates or two dimensionally on a photographic plate. Both methods of capture involved optical post processing to yield results which were qualitative only.

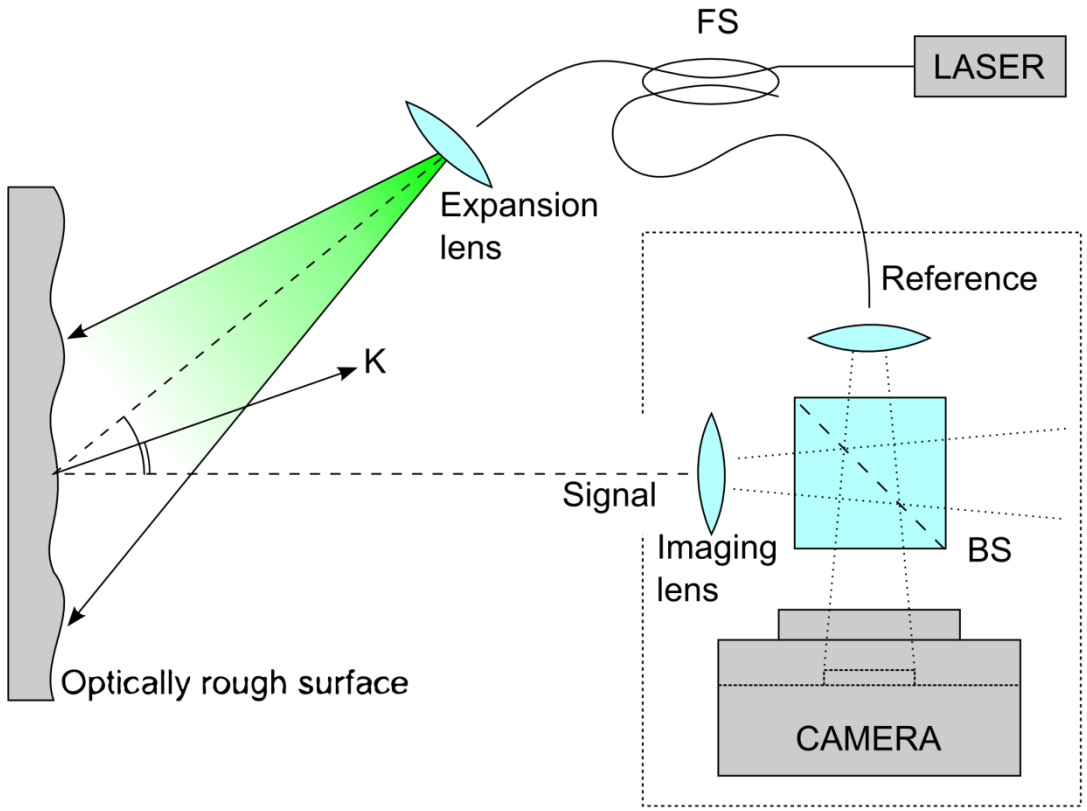


Figure 3-2: Single component ESPI with fibre delivered reference beam. (BS: Cube beamsplitter, FS: fibre splitter (directional coupler), k: sensitivity vector)

Later adoption of electronic image sensors (CMOS and CCD arrays) lead to the development of the now established technique of electronic speckle pattern interferometry (ESPI). This technique is also termed TV holography in earlier papers due to the similarities between it and holographic techniques. The principle remains similar, with an imaging beam and a reference beam, though variations exist on the placement of the reference beam. In a single illumination beam ESPI system the object beam is incident on the test object, which is then imaged onto an electronic imaging sensor through a beamsplitter. Into one arm of the beamsplitter is directed the reference beam, which is sampled from the object beam and can be delivered either through air or by way of optical fibres, as is shown in Figure 3-2. The reference beam is, in this case, a non-speckled, uniform phase wavefront that is transmitted through the beamsplitter onto the imaging sensor to transduce phase changes in the object beam into intensity variations. The out of plane displacement (normal to the surface), w, is related to the phase change by the following [18]:

$$w = \frac{\Delta\theta}{2\pi}\frac{\lambda}{2} \qquad 3\text{-}2$$

A dual beam ESPI system, such as that shown in Figure 3-3, does not have an internal reference channel, instead illuminating the object simultaneously with two beams. This produces two overlapping speckle fields that interfere. If the beams are collimated and angled to be incident on the surface at equal and opposite angles, the sensitivity of the device is constrained to in-plane displacements, given by equation 3-3 [19], where ϕ is the measured phase, λ the illumination wavelength, α the illumination angles to surface normal, and u the in-plane displacement.

$$\Delta\phi = \frac{4\pi u \sin\alpha}{\lambda} \qquad 3\text{-}3$$

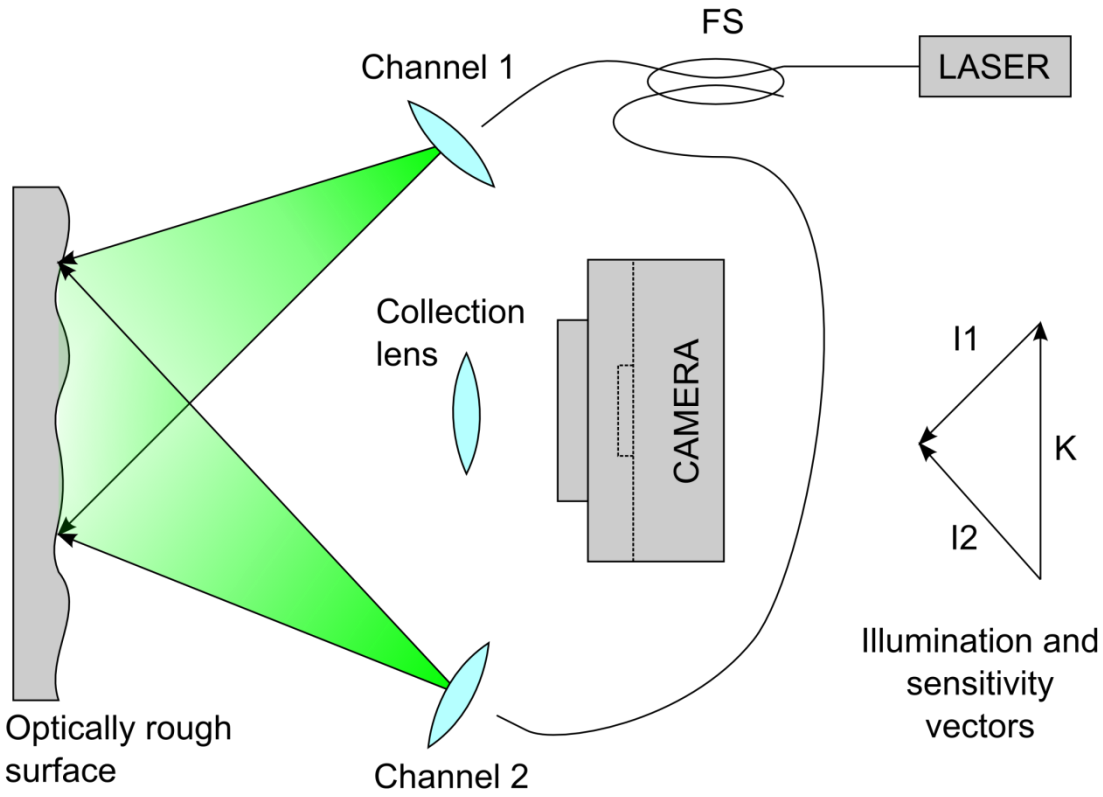


Figure 3-3: In-plane ESPI, using a fibre splitter to split a source between two channels, delivered to the surface by fibre via expansion lenses. (FS: Fibre splitter, I1 and I2: Illumination vectors, k: Sensitivity vector (subtracting I1 from I2)

3.2.4 Shearography

Shearography is a technique sensitive to full-field displacement gradients across an object subject to a load induced deformation, and in effect performs an optical

differentiation of surface displacement fields. In this method, the surface of the test object is illuminated by the output from a laser to produce the speckle pattern and is imaged through a shearing device onto a digital image sensor.

The shearing device is typically a common-path interferometer that allows the image to be split into two identical images, displaced relative to one another and recombined into a single image on the image sensor. This allows the light scattered from a point on the surface to interfere with the light scattered from a neighbouring point. On deforming the surface, these points in the images will move relative to each other, causing the phase relationship between the rays to change, modulating the intensity of the speckle at that point of the image. Comparison of two speckle patterns, taken before and after loading the object, allows the change in phase to be calculated through the use of phase stepping algorithms. These are discussed in more detail in section 3.4. Due to the use of an image shearing device, shearography can be said to be self-referencing, in comparison to ESPI which has an external reference. Shearography is therefore more resilient to vibration.

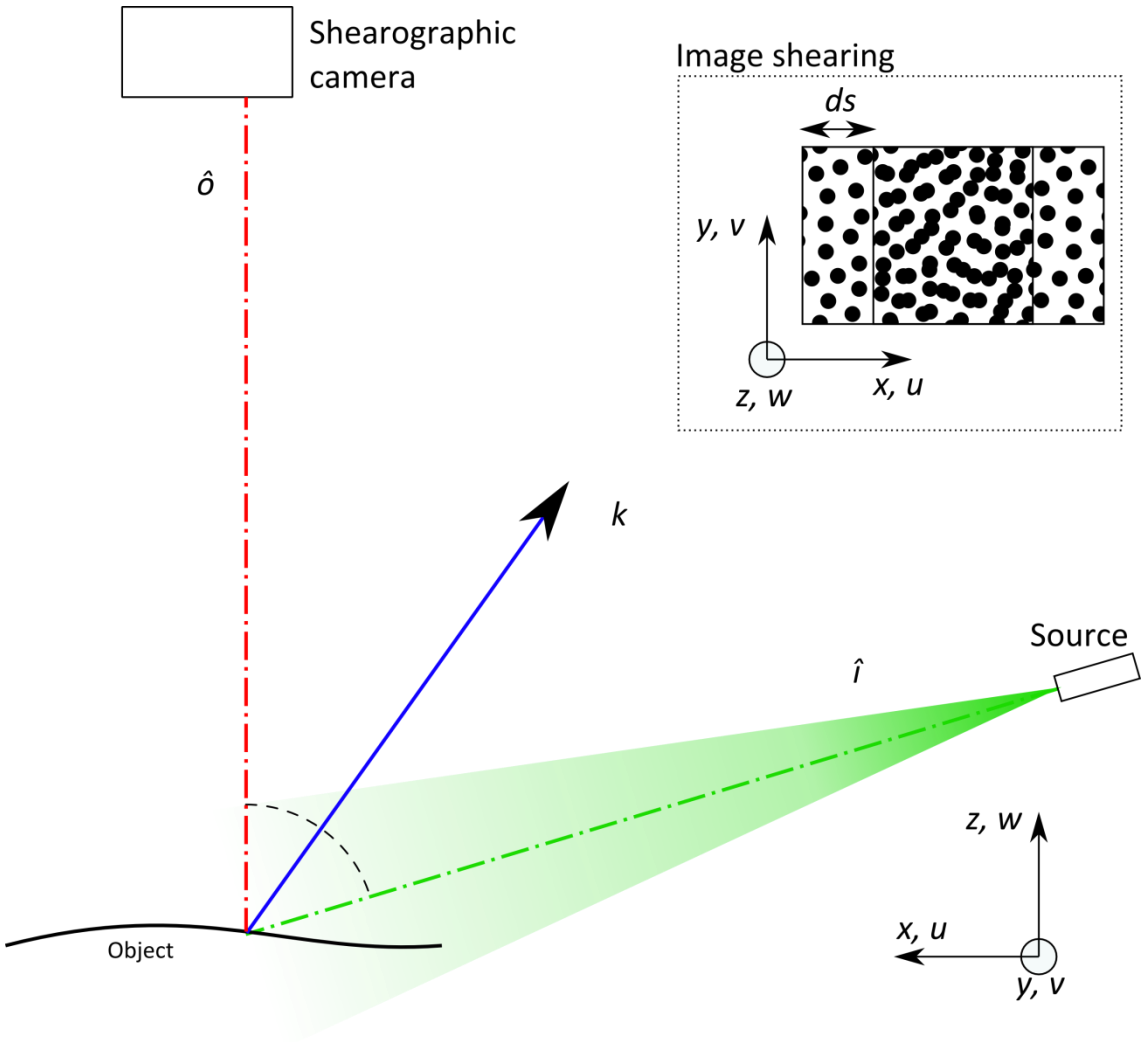


Figure 3-4: A simple shearography system for providing a qualitative map of the surface strain experienced by an object subjected to loading. The system uses a single laser source for illumination along ***i****, an image shearing camera (shearographic camera) observing along* ***o****, and is most sensitive to displacements gradients in the direction of the sensitivity vector* ***k****. Loading the object induces displacements on the surface of the object which manifest as displacement gradients in u, v, and w which correspond with the x, y, and z directions when measured with the shearography. Inset: illustrates image shearing, showing how the shearographic camera laterally shears the images by an amount ds in the x,y-plane to overlap and interfere neighbouring speckles. An example of a shearographic camera can be seen in Figure 3-5(a).*

In general, the specific subset of surface displacement gradients that a system will detect is determined by the shear direction induced in the image. The sensitivity of the measurements depend on the shear magnitude, δx or δy. Tensile (one dimensional) strain in an object can be expressed as $\varepsilon = \Delta L/L$ [20], which corresponds closely to the displacement gradients encoded in the phase change maps. The phase changes

induced between the loaded and unloaded states of an object for shearography are shown below for x- and y-shears respectively (where $\Delta\phi \equiv \phi$ in some literature):

$$\Delta\phi_x = \frac{2\pi\delta x}{\lambda}\left[\frac{\delta u}{\delta x}k_x + \frac{\delta v}{\delta x}k_y + \frac{\delta w}{\delta x}k_z\right] \qquad 3\text{-}4$$

$$\Delta\phi_y = \frac{2\pi\delta y}{\lambda}\left[\frac{\delta u}{\delta y}k_x + \frac{\delta v}{\delta y}k_y + \frac{\delta w}{\delta y}k_z\right] \qquad 3\text{-}5$$

For small shears, equations 3-4 and 3-5 approximate to:

$$\Delta\phi_x = \frac{2\pi\delta x}{\lambda}\left[\frac{\partial u}{\partial x}k_x + \frac{\partial v}{\partial x}k_y + \frac{\partial w}{\partial x}k_z\right] \qquad 3\text{-}6$$

$$\Delta\phi_y = \frac{2\pi\delta y}{\lambda}\left[\frac{\partial u}{\partial y}k_x + \frac{\partial v}{\partial y}k_y + \frac{\partial w}{\partial y}k_z\right] \qquad 3\text{-}7$$

Where the partial derivatives of u, v and w with respect to x and y are the displacement gradient components, and the terms in k are sensitivity vector components in x, y, and z. The sensitivity vector is dependent on the illumination and observation point locations and is the bisector of the angle between them.

The differential terms can be related linearly to the surface tensile strain as follows:

$$\varepsilon_{xx} = \frac{\partial u}{\partial x};\ \varepsilon_{yy} = \frac{\partial v}{\partial y} \qquad 3\text{-}8$$

The linear relationship to surface shear strain is similarly:

$$\gamma_{xy} = \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \qquad 3\text{-}9$$

A shearographic measurement therefore could contain a combination of in- and out-of-plane strain components, and separation of these is discussed in more detail in section 3.4.4. Multi-component configurations of shearography systems are discussed in section 3.3. The contributions of the various surface displacement gradients to a phase map can by modified by changing the sensitivity vector, $\boldsymbol{k}$, by changing the illumination and/or viewing configuration. This allows a shearography system to be configured to select various surface displacement gradient combinations, a technique taken from three-dimensional ESPI [14]. The expressions relating phase with the displacement gradient for arbitrary illumination and/or observation points are as follows for shear applied in the x and y directions respectively [21]:

$$\Delta\phi_x = \frac{2\pi\delta x}{\lambda}\left[sin\,\theta_{xz}\,cos\,\theta_{yz}\frac{\partial u}{\partial x} + cos\,\theta_{xz}\,sin\,\theta_{yz}\frac{\partial v}{\partial x} + \left(1 + cos\,\theta_{xz}\,cos\,\theta_{yz}\right)\frac{\partial w}{\partial x}\right] \qquad 3\text{-}10$$

$$\Delta\phi_y = \frac{2\pi\delta y}{\lambda}\left[sin\,\theta_{xz}\,cos\,\theta_{yz}\frac{\partial u}{\partial y} + cos\,\theta_{xz}\,sin\,\theta_{yz}\frac{\partial v}{\partial y} + \left(1 + cos\,\theta_{xz}\,cos\,\theta_{yz}\right)\frac{\partial w}{\partial y}\right] \qquad 3\text{-}11$$

The sensitivity of a shearography system is dependent on the magnitudes of the sensitivity vector components, as can be seen in equations 3-10 and 3-11 for a small field of view, and more generally for the components as defined in 3-12.

$$k_x = \boldsymbol{k} \cdot \hat{\boldsymbol{l}}$$

$$k_y = \boldsymbol{k} \cdot \hat{\boldsymbol{m}} \qquad 3\text{-}12$$

$$k_z = \boldsymbol{k} \cdot \hat{\boldsymbol{n}}$$

Where $\hat{\boldsymbol{l}}$, $\hat{\boldsymbol{m}}$, and $\hat{\boldsymbol{n}}$ are unit vectors along the x, y, and z axes respectively.

It can be seen therefore that, by careful positioning of the illumination sources, the instrument can be configured to be insensitive to various surface displacement gradient components [14]. For example, for an illumination source in the xz-plane, the phase map for an x-sheared image would be insensitive to the component $\partial v/\partial x$ and can be described as:

$$\Delta\phi_x = \frac{2\pi\delta x}{\lambda}\left[sin\,\theta_{xz}\frac{\partial u}{\partial x} + (1 + cos\,\theta_{xz})\frac{\partial w}{\partial x}\right] \qquad 3\text{-}13$$

In order to measure stress components in multiple axes, sequential measurements with the shear direction changed between measurements can be taken, or a method of multiplexing multiple shear directions must be found in order to measure each simultaneously. In section 3.3, various multi-component shearography configurations are discussed. Methods for processing the images to determine the strain are discussed in section 3.4.

3.3 Implementations

3.3.1 Strain measurement

A single channel shearography system, shown in Figure 3-5 (a), consists of a laser source that is used to illuminate the surface of an object being tested, and an imaging head consisting of an image-shearing device and an image sensor, collectively referred to here as the shearographic camera (SC). Speckle patterns resulting from illumination of the object's surface before and after loading are imaged onto the sensor and are processed to reveal the phase change, $\Delta\phi$, imposed on the scattered wavefront due to the surface deformation gradients.

The phase change is described in equation 3-14 and shows that the phase is related to the applied image shear, δs, which can be in either the horizontal ($\partial s = \partial x$) or vertical ($\partial s = \partial y$) direction.

$$\Delta\phi_s = \frac{2\pi\delta s}{\lambda}\left[k_x\frac{\partial u}{\partial s} + k_y\frac{\partial v}{\partial s} + k_z\frac{\partial w}{\partial s}\right] \qquad 3\text{-}14$$

Therefore, a phase map retrieved from a single channel generally contains contributions from the two in-plane and one out-of-plane displacement gradient terms. From measurements undertaken using a single channel system, these orthogonal terms are inseparable and so the phase map is generally useful for obtaining only a qualitative indication of the presence and locations of defects within an object, as indicated by a local change in the phase map, and this is the application for which shearography is most used [11].

Shearography was first reported in 1973 by Leendertz and Butters [22] in the study of bending moments, terming the technique "image shearing speckle pattern interferometry". A Michelson interferometer was used to laterally shear two laser speckle images and reveal the correlation fringes resulting from a bending load applied to the object. This was developed as an improvement to using speckle pattern interferometry, in which, due to the sensitivity of ESPI to displacement, the fringe patterns had to be double differentiated to reveal the relationship to the bending on the surface, increasing the effects of error in the measurements. Image shearing allowed the first differentiation step to be made optically, reducing the number of calculations needed. Another early generation of shearography came nearly ten years later, in 1983, when a system used to study surface profile was reported by Hung [23]. It used a conventional camera with a double aperture, one of which was covered by a glass wedge to produce a sheared image. This system was able to resolve fringes through a Fourier transform process utilising a double exposed image on a photographic plate. The sensitivity of the camera was fixed in the device by the optical arrangement but could be controlled externally varying the refractive index around the test object between exposures to produce low visibility moiré fringes. Hung later reported the application of shearography to the field of non-destructive testing to locate defects in products during manufacture [24], a field to which the technique is well suited [25].

Quantitative measurement of full surface strain requires the use of at least three measurement channels, each sensitive to a different combination of the components of surface displacement gradient. This was first demonstrated by Steinchen et al. in [26] for a three-illumination channels system. In quantitative shearography, the detectable

strain components are obtained through processing phase maps from each measurement channel, which by definition (equations 3-10 and 3-11) contain contributions from multiple surface strain components. Information obtained from multiple channel phase measurements allows the surface displacement gradient terms to be separated, and, with phase maps captured from these channels with the orthogonal shear direction applied, quantitative measurement of the surface strain components as described by the strain tensor, *S*, in equation 3-35 can be achieved. Using a system of multiple illumination or observation directions, similar to the two configurations shown in Figure 3-5, parts (b) and (c), it is possible to obtain quantitative measurement of the three-dimensional surface strain. The channels can be placed around a central point to obtain the result, as per Steinchen [14] and can be optimised to simplify the calculation by performing a matrix transformation on channels placed at three corners of a square arranged around a central observation point [27], reducing the calculation to a matter of addition and subtraction of pairs of phase maps. Errors may be reduced through the addition of an additional fourth channel, giving a redundancy in the results which can be exploited to remove low quality measurement regions [28] (see more on this in section 3.4.4).

The first demonstration of quantitative full three-dimensional determination of surface strain in the DEP was reported by James and Tatam in 1999 [27][29], using techniques developed from the field of quantitative ESPI. Temporal phase stepping was employed using a path imbalanced Michelson image-shearing interferometer and laser wavelength changes induced through drive current modulation. In order to measure the full surface strain, the experiment needed to be repeated with the same deformation but with the shear direction switched between measurements. Information was recorded from each channel using time-division multiplexing to capture the phase stepped speckle images from each source in sequence.

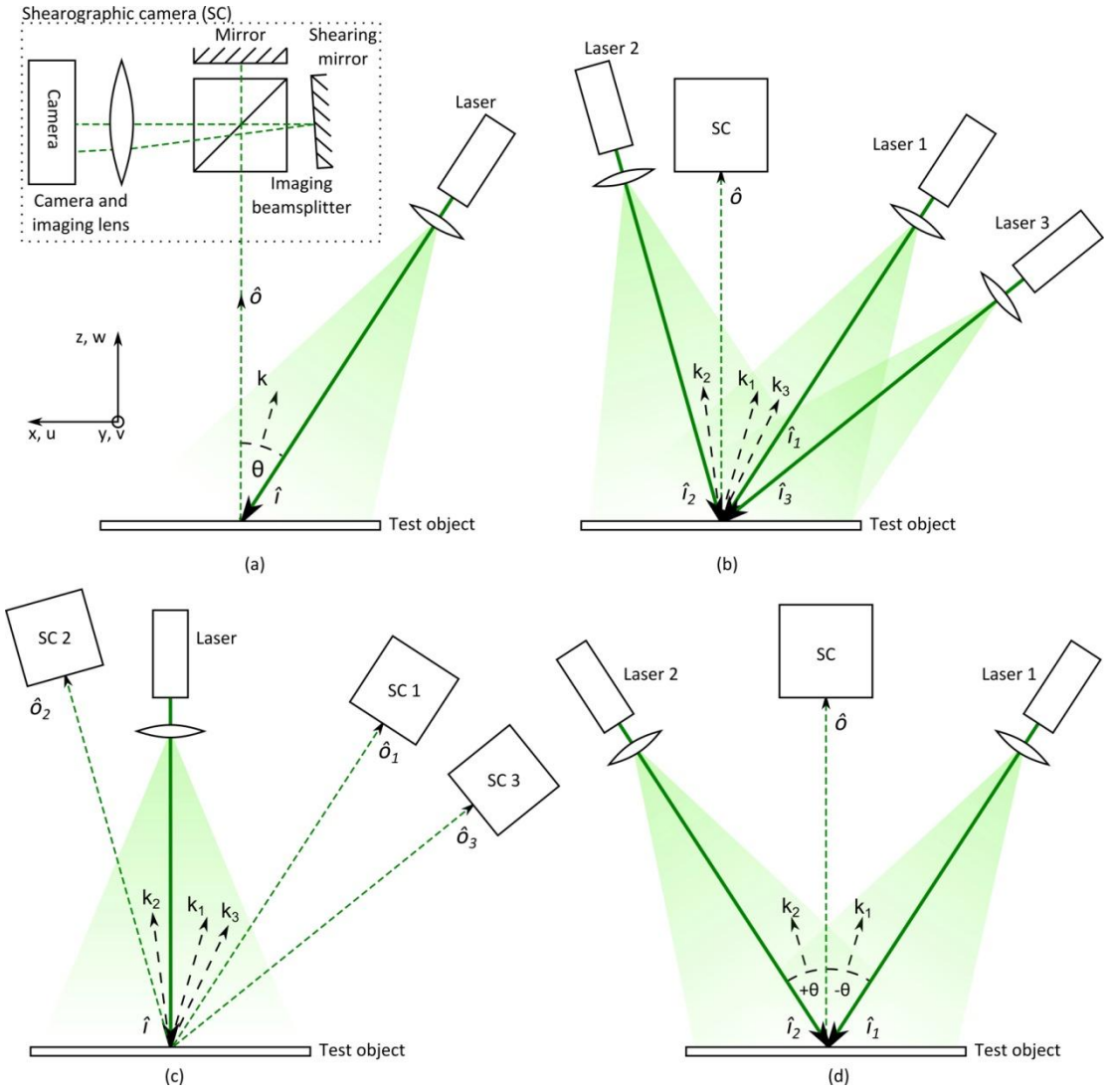


Figure 3-5: (a) Representation of a single channel shearography system and the imaging optics of the shearographic camera (SC, dotted rectangle) [23]; (b) a three channel system using multiple illumination directions, î, around a single shearographic camera to vary the sensitivity [26]; (c) three channel system using multiple observation directions, ô, formed of multiple cameras around a single laser [30]; (d) a two channel in-plane system with two lasers arranged at equal angles around the observation axis [14].

A configuration that can be used to measure the out-of-plane displacement gradient component and one of the in-plane displacement gradient component is shown in Figure 3-5(d) [31]. Addition and subtraction of phase maps, obtained from two channels positioned symmetrically in a plane defined by the observation and illumination vectors, yields the out-of-plane ($\partial w/\partial s$) and in-plane ($\partial u/\partial s$) displacement gradient components. For this configuration, the displacement gradient components can be calculated using:

$$\frac{\partial u}{\partial s} = \frac{\lambda}{4\pi\delta s k_x}(\Delta\phi_+ - \Delta\phi_-) \qquad 3\text{-}15$$

$$\frac{\partial w}{\partial s} = \frac{\lambda}{4\pi\delta s k_z}(\Delta\phi_+ + \Delta\phi_-) \qquad 3\text{-}16$$

A variant of this technique is based on TV-holography, where the two measurement channels are imaged through the shearing interferometer as they illuminate simultaneously the surface of the object. This produces a low contrast Moiré fringe pattern of the in-plane strain, but is not a quantitative technique [14].

Therefore, in order to obtain quantitative measurements of surface strain, more than one measurement channel must be employed, multiplexed using methods discussed in section 3.3.3

3.3.2 Slope measurement

Shearography also can be used in the measurement of surface contours. This is useful in itself for characterising a surface, but also has an application in reducing the error in phase measurements that arises from the assumption of a constant image shear across the field of view. This assumption is erroneous when the surface being measured is not planar. As the distance to the surface distance changes across the field of view, the apparent slope of the object changes. Groves et al. developed a shearography system to make full surface strain measurements from a curved surface where the surface slope was characterised using the same instrument, via source displacement, and the slope measurement used to determine the image shear across the field of view [32].

Slope measurements can be made in shearography by variation of either illumination wavelength, illumination point, or the refractive index of the medium surrounding the object, respectively called double-wavelength, two-point source and double-refractive index surface profiling. These can be respectively related to the phase by the following expressions:

$$\Delta\phi = \frac{2\pi L n}{\lambda^2}(\delta\lambda) \qquad 3\text{-}17$$

$$\Delta\phi = \frac{2\pi L}{\lambda}(\delta n) \qquad 3\text{-}18$$

$$\Delta\phi = \frac{2\pi n}{\lambda}(\delta L) \qquad 3\text{-}19$$

In these expressions, L is a geometric relation between the illumination point (or points), the points on the surface, and the final positions on the image sensor.

These techniques require paired images to obtain the slope fringes. Those from the double-wavelength or the two-point source method are potentially likely to benefit from a multiplexing scheme, such as FDM, that allows the capture of the two images simultaneously.

It is possible to design systems using methods that allow phase maps to be acquired experimentally without requiring phase discontinuities to be considered, such as in the case where phase stepping might be employed. One such approach is the equivalent wavelength method, where two sources closely spaced in wavelength are used to illuminate the surface of an object that is to be deformed by an amount that is expected to not exceed the difference between the two source wavelengths. This allows the phase to vary continuously over the deformation region without discontinuities from the phase change exceeding 2π radians.

Another approach is to vary the sensitivity vector between exposures to produce a double image similar to that in shearography, but in this approach the phase maps are sheared through a physical movement of the source over a few degrees. This produces phase gradient maps that correspond closely to the displacement gradient measured in shearography. This approach has the disadvantage of increased noise [2].

The double wavelength and two-point source approaches have been shown to have applications in producing phase maps from surfaces that do not need unwrapping using the phase derivative method[15]. More conventionally, the double wavelength technique can be applied in such a way that the equivalent wavelength (equation 3-20 [11]) created from two wavelengths at λ_1 and λ_2 can be selected to be longer than the expected height deviation in the surface.

$$\lambda_{eq} = \frac{\lambda_1\lambda_2}{|\lambda_1 - \lambda_2|} \qquad 3\text{-}20$$

This means that when the phase is determined from the equivalent wavelength method to measure the surface slope, there is no wrapping of the phase, and hence no phase unwrapping algorithm is required. However, it has also been shown that the noise in the system increases as the equivalent wavelength increases. The two-point source method can be used similarly, producing phase maps that can be differentiated to produce slope information from wrapped phase maps, the final result, after

differentiating the sine and cosines of the phase, being continuous. Interestingly, the technique was also demonstrated in the same paper [15] with Moiré fringes on a discontinuously sloped surface consisting of steps, and the technique was able to evaluate the phase at each level of the stepped structure without ambiguity.

3.3.3 Multiplexing in speckle interferometry

The advantage of the use of multiplexing schemes in shearography systems lies in reducing the time taken to capture the full set of images required to derive the full surface strain. For example, the use of equation 3-40 requires the production of six phase maps. If these are to be captured using a five-frame phase stepping algorithm, then a total of thirty image capture events are required. Using time-division multiplexing (TDM) to obtain these requires switching between channels and then measuring the light incident on the sensor. Steinchen reported a system which relied on this approach in [26], and many other developments in shearography still rely on TDM in some form to move between data capture events. However, in this chapter, FDM is employed as a scheme for multiplexing multiple illumination channels together on a single camera. Other schemes for achieving multiplexing of various stages of the capture process have been reported, and these are reviewed here.

The use of wavelength division multiplexing to capture the speckle patterns from three channels in a 3D-shearography system has been reported [33]. The interferometer is shown in Figure 3-6. Three sources, operating at close but distinct wavelengths, were used to illuminate an object from three different directions. The speckle patterns were recorded on three CCD cameras simultaneously, located behind wavelength and polarisation filters to isolate one channel per camera. This had the effect of reducing capture time by a factor of three, however it did not allow the simultaneous capture of information from two shear directions. The multiple wavelength approach can be adapted to FDM, which could also facilitate shear multiplexing.

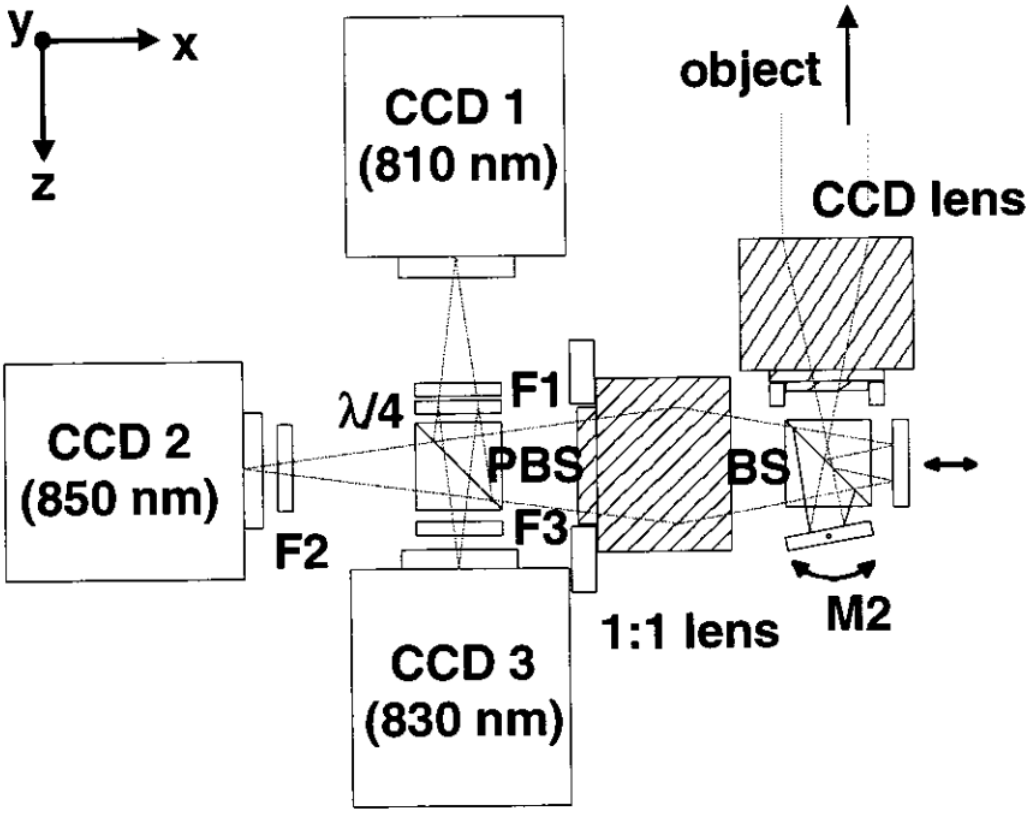


Figure 3-6: Wavelength-division multiplexing applied to shearography. Uses three wavelengths separated which are each split into the three channels in the imaging head and the wavelengths separated in each channel by way of a wavelength selective filter. F1-3: Channel filters; PBS: Polarising beam splitter; BS: Beam splitter; M2: Shearing mirror. Figure reproduced from [33].

One approach is to capture images from multiple channels simultaneously, affording the advantage of only requiring a single source to illuminate the object. This can be achieved through the use of multiple image shearing interferometers and cameras at each observation point, arranged around the central illumination point. This approach was reported in [30]. Speckle patterns may be imaged from each channel onto different sub-regions of the imaging sensor, as explored in [34] using coherent imaging fibre bundles arranged around the corners of a square centred on a single illumination point. The operation of the system was demonstrated by monitoring a rotating PTFE plate thermally loaded by heating an aluminium disc attached to the rear of the plate. The laser pulses and the camera frame rate were synchronised to a trigger signal coming from the rotation stage onto which the plate was attached. This allowed the light pulses to effectively freeze the rotation of the plate in each frame, removing motion blur and ensuring the image frames were captured at the same rotational position. The advantage is the ability to capture all the channels simultaneously, though image resolution is sacrificed by using only a small fraction of the sensor for

each channel. Combined with the spatial carrier fringe method, this allowed the capture of four channels of data, with all the information required to determine the phase in a single image capture event with a single image shear direction, during a single pulse. The process would have to be repeated with the interferometer rotated about the observation axis to get the data from the orthogonal shear with that configuration.

With distributed observation points, the view from each is different, resulting in perspective distortion on each channel. This needs to be corrected using image dewarping algorithms and then the images need re-centred on a common centre to allow the images to be processed [35]. It is possible to multiplex full-sensor images, rather than distributed sub-regions as per the coherent fibre bundle approach, using FDM to capture multiple views, though the benefits in this situation may be minimal, and correction of perspective distortion would still be necessary.

Polarisation-division multiplexing [10] was used to multiplex the orthogonal image shears with each phase step. Achieved through laser drive current modulation with birefringent fibres, it was possible to use the current modulation to control the polarisation state of the light emitted at the birefringent fibre end, and, by using a path length imbalanced shearing interferometer, to facilitate temporal phase stepping. Coupling the light equally into both the fast and slow axes of the fibre allowed the small changes in wavelength, induced by controlling the drive current on the laser diode, to result in a change in polarisation state on exiting the fibre due to the effect of the different phase velocities in the two axes. Polarisation control of the illumination allowed the use of polarisation sensitive components in the imaging head to switch the applied image shear optically, using the polarisation state of the scattered light as the switch. The interferometer is shown in Figure 3-7. There was a 20% depolarisation of the light on scattering however, introducing cross-talk between the measurements made with the two orthogonal shears. If the polarisation states were to be multiplexed using FDM, then the number of steps needed per measurement could be halved.

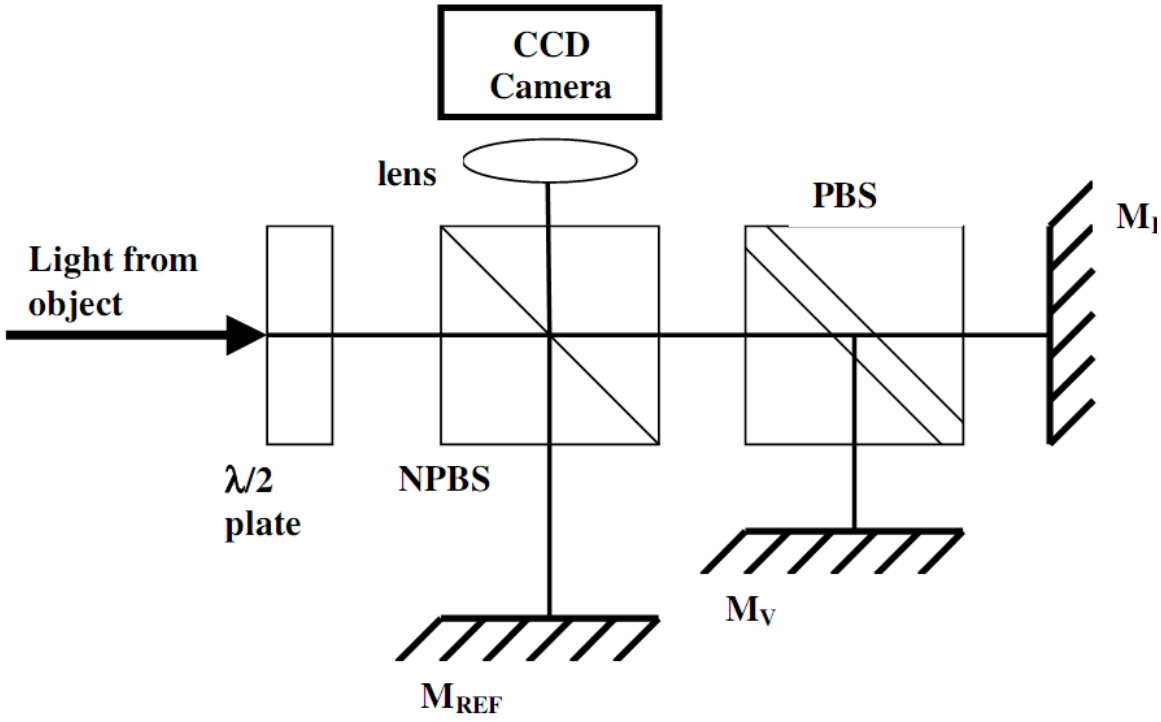


Figure 3-7: The imaging head used in polarisation-division multiplexing. NPBS: Non-polarising beamsplitter; PBS: Polarising beam splitter; M_H: Horizontal shearing mirror; M_V: Vertical shearing mirror, M_{REF}: Reference wavefront mirror. Figure reproduced from [10].

3.4 Image processing

3.4.1 Intensity to phase – temporal phase stepping

Speckle interferograms recorded by a digital camera are two-dimensional, digitised intensity maps of the light incident on the sensor. The conversion of the light field to intensity on a digital camera array causes a loss of the complex amplitude of the light. In order to use the interferograms obtained in this manner quantitatively, consideration must be given to extraction of phase information from the interference patterns.

The intensity of an interference pattern can be represented with respect to the background intensity, I_0, visibility, γ, and the phase by the following [9]:

$$I = I_0(1 + \gamma \cos \phi) \qquad 3\text{-}21$$

However, recording in this manner causes the phase information contained in the speckle patterns to be lost. Regaining this information requires the use of algorithms that convert intensity into phase. As phase can take both positive and negative values,

the resulting phase images must be mapped to a suitable colour or grey scale before they can be displayed as an image. The resulting phase images are often termed *wrapped phase maps*, as the phase is constrained between within a range of $-\pi$ to π radians.

Methods for extracting phase information from a speckle interference pattern can be classified into two types: temporal and spatial [4]. Another range of methods exist for the extraction of information directly from the intensity pattern in a single image [36] but this is considered less accurate, by up to several orders of magnitude[4]. Spatial and temporal methods encode the recorded information with phase information that can be extracted by appropriate techniques.

There are many methods used to encode an interference pattern with phase information. In the temporal regime, a series of images are usually recorded where the path length imbalance in the shearing interferometer is varied consistently between frames by a known amount, leading to a phase term α being introduced into equation 3-21:

$$I = I_0[1 + \gamma\, cos(\phi + \alpha)] \qquad 3\text{-}22$$

Where $\alpha = 2n\pi$ and is the relative phase change induced in the instrument, with n corresponding to the desired fraction of 2π phase at each step. Two approaches can be taken to apply the phase shift, either constant ramping of a phase controlling element whilst taking readings and integrating the results (the so-called "integrating bucket" method) [4], or discrete steps may be taken to a desired phase and measurements taken at each step (phase stepping) before using a phase stepping algorithm to calculate the phase. A reduction in fringe visibility due to the requirement to measure over a time period while the fringes are moving introduces uncertainty into results obtained through phase ramping [4].

What follows in this section is a short review on temporal phase stepping algorithms.

A variety of phase stepping algorithms exist for extracting phase information across a wavefront using a series of speckle interference pattern image frames with phase steps between them. In 1966 Carré [37] presented an influential paper detailing a method determine the phase from a series of intensity measurements with a constant phase step applied to the wavefront between each measurement, now called the Carré technique, which is shown below in equation 3-23. Modern techniques apply this algorithm across the full field images of the pattern, but originally it was demonstrated on intensity variations at a single point.

$$tan\,\varphi = \frac{\sqrt{[(I_1 - I_4)(I_2 - I_3)][3(I_2 - I_3)(I_1 - I_4)]}}{(I_2 + I_3)(I_1 + I_4)} \qquad 3\text{-}23$$

This technique did not rely on a known phase step, but rather a constant step between measurements, the value of which could be later deduced from the experimental data [19].

A four frame method (equation 3-24) was presented by Wyant in 1982 [38] for $\pi/2$ phase steps at $\alpha = 0, \pi/2, \pi,$ and $3\pi/2$.

$$tan\,\varphi = \frac{I_4 - I_2}{I_1 - I_3} \qquad 3\text{-}24$$

In 1993, Creath wrote [4] that a minimum of three images of the interference pattern would provide enough information to extract the wavefront phase. A three frame algorithm (Equation 3-25) was presented for use with $\pi/2$ phase steps at $\alpha = \pi/4, 3\pi/4,$ and $5\pi/4$.

$$tan\,\varphi = \frac{I_3 - I_2}{I_1 - I_2} \qquad 3\text{-}25$$

A three step algorithm for an arbitrary phase step α was also given (equation 3-26).

$$tan\,\varphi = \left(\frac{1 - cos\,\alpha}{sin\,\alpha}\right)\frac{I_1 - I_3}{2I_2 - I_1 - I_3} \qquad 3\text{-}26$$

Schmit and Creath [39] demonstrated that errors in the calculated phase could be reduced significantly by incorporating an appropriately shaped averaging function across the image set in the algorithm derivation. Bell-curve averaging was shown to reduce errors in phase calculations to below the level of noise uncertainty. The 8-BELL7 algorithm (equation 3-27) for use with $\pi/2$ phase steps was suggested as a more tolerant algorithm than the Schwider-Hariharan method (equation 3-28) [40][41].

$$tan\,\varphi = \frac{I_1 + 5I_2 - 11I_3 - 15I_4 + 15I_5 + 11I_6 - 5I_7 - I_8}{I_1 - 5I_2 - 11I_3 + 15I_4 + 15I_5 - 11I_6 - 5I_7 + I_8} \qquad 3\text{-}27$$

$$tan\,\varphi = \frac{2(I_2 - I_4)}{2I_3 - I_5 - I_1} \qquad 3\text{-}28$$

Equation 3-28 applies only for the case of applied phase steps of $\alpha = \pi/2$, otherwise the more general case shown in equation 3-29 may be applied for a known phase step.

$$tan\,\varphi = \left(\frac{1 - cos\,2\alpha}{sin\,\alpha}\right)\frac{2(I_2 - I_4)}{2I_3 - I_5 - I_1} \qquad 3\text{-}29$$

Angel and Wizinowich [42][43] proposed a method for quickly obtaining the phase shifted images so as to reduce the effects of vibrations on the measurements. This was called the 2 + 1 algorithm (equation 3-30).

$$tan\,\varphi = \frac{I_2 - I_C}{I_1 - I_C} \qquad 3\text{-}30$$

In this, I_C is a composite image that gives the background DC intensity, and is considered relatively independent of the other two images such that it can be recorded at a different time during the experiment.

In order to obtain phase values from the equations presented, the arctan has to be taken. This will produce errors in the results depending on the actual value of the phase, as the arctan only considers phase values in the range *0* to π but the speckle field contains phase values modulo 2π. In order to remove this source of error, the signs of the numerator and denominator need to be considered in order to work out which quadrant the phase lies in and adjust the result accordingly. In programming this is easily accomplished with the *atan2* or *arctan2* function, which does this automatically.

The scanning phase shift technique reported by Vikhagen allows a buffer of many images to be analysed when an unknown or random phase shift is present in the system [44].

$$cos\,\varphi = \frac{I_i - I_0}{\gamma I_0} \text{ for the } i\text{th frame} \qquad 3\text{-}31$$

In this method, the background intensity is given by $I_0 = (I_{max} - I_{min})/2$ and the visibility is obtained by $\gamma = (I_{max} - I_{min})/(I_{max} + I_{min})$, both of which are easily obtained from the image bank.

In the work presented in this thesis, the Schwider-Harriharan 5-step method was chosen due its robust nature in dealing with non-linearity in the detector. Higher order terms in the intensity conversion at the detector, such as might be expected with the application of the FDM algorithm that omits the square-root step, were shown [4] not to affect the phase when present in the intensity measurements.

3.4.2 Phase unwrapping

A phase map produced by a phase stepping algorithm has phase discontinuities, locations where the phase values take a step change as a consequence of using an inverse trigonometric function, usually the arctangent, to determine the phase. These phase maps are considered to be wrapped, as their values constrained within a range of 2π. It will be necessary to perform phase unwrapping to obtain the true phase relationship between points in the image from a wrapped phase map. This is where

phase unwrapping is employed, the techniques of which are briefly reviewed in this section.

In the practical work of this chapter, the phase is unwrapped using freely licensed code from Liverpool John Moores University, which uses the iso-phase method [45] to unwrap the phase by regions. This approach is described in this section, along with a review of other methods available.

The simplest form of phase unwrapping is Itoh's one dimensional method where the phase difference Δ between a point n and the $n+1$ point is added to the phase value of the point n in a row, as per equation 3-32:

$$\phi_n = \phi_{n-1} + \Delta_{n-1} \qquad 3\text{-}32$$

This can then be applied row by row incrementally to unwrap the phase of whole image. Care must be taken around regions of large phase change, where $\Delta_{n-1} > \pi$, or where there is a high fringe density, as aliasing may occur, and as such it is suggested that the application of the Itoh method be taken as an estimate of the unwrapped phase. As noise in the signal increases, so discontinuities will appear in the unwrapped phase.

Phase residues are the biggest problem for phase unwrapping. A simple unwrapping algorithm like that of Itoh's method is best utilised on noise-free fringe maps without fringe discontinuities in the wrapped data, as these will produce errors that will then propagate to the end of the rows [46]. It is possible to process the phase maps to reduce the noise level present prior to unwrapping, though noise reduction needs to be offset against the requirement to maintain fringe edge definition. Various convolution filters can be used to improve the performance of phase unwrapping, and these are discussed in section 3.4.3. Noise reduction will reduce the likelihood of errors being introduced in one dimensional phase unwrapping, though it is not possible to completely remove noise and some may still remain, especially at the edges of phase maps where the illumination intensities may be lower. With a one-dimensional process initialised in the noisy edge regions by reference to the first point of each row, it is quite likely that each unwrapped row will be out of phase with those adjacent. A method attributed to Takeda [2] involves applying a further 1D unwrap down the central column of pixels to allow the error to be removed and restore the phase relationship between lines.

Another method from Robinson and Williams unwraps using a one dimensional algorithm, but includes a check on the value of each pixel being unwrapped relative to two previously unwrapped pixels, namely the one behind and one on the adjacent

unwrapped line [47]. If the unwrapped value of the phase for that pixel does not fit with the reference pixels, or is bad data, then the value may be masked to reduce error propagation effects. There is an issue to be considered when choosing the first line to unwrap, as if this contains errors, it can then influence the rest of the unwrapping process. In top-to-bottom unwrapping, the first unwrapped line would be taken from the higher noise region at the edge, so increasing the risk of error. Alternatively, unwrapping can begin in the higher signal region in the centre of the wrapped phase map to reduce the likelihood of errors in the first line propagating throughout the rest of the map. The scan can then proceed in a linear fashion as before, radially, or may spiral outwards as per Vrooman and Maas [47], allowing pixels to be checked against an average of three others.

Bad pixels are relatively easily compensated for by a pixel referencing approach, but if the defect is larger than a single pixel, creating a hole in the phase map, the processing complexity is increased greatly. Apart from Vrooman and Maas, the algorithms presented so far do not unwrap correctly across certain defect shapes and so some effort must be made to compensate for this. Huntley approached the problem with a spiralling pixel referencing method that iteratively filled these defect regions from the edges inwards so that unwrapping could be applied [47]. Multiple passes of this filtering approach may be needed if the hole is cut by the edge of the map, slowing the process down.

It is possible to use a pixel queuing method that unwraps phase maps according to the quality of the individual pixels. By comparing a pixel to neighbouring pixels to find the lowest phase gradient, the phase map can be ordered into good and bad data. From this concept, Schorner analysed the entire map and divided it into paths that the unwrapping algorithm will follow between good data points using the minimum spanning tree principle [48]. The method devised by Kwon does not have a set path to follow, rather it follows dynamically the highest quality path it can from a starting pixel of good data [49]. Both these methods work by unwrapping the best quality data first so as to reduce error effects from bad regions propagating into good. Good data can also be defined as a point where the fringe contrast is high. Good data therefore occurs in low noise regions away from the fringe edges in the wrapped phase maps and bad data appears at fringe edges and in regions of high noise. By careful selection, regions of noise can be removed from the pixel queue [50].

Goldstein's algorithm is a popular choice for phase unwrapping due to the robust nature of the algorithm to phase residues (point-like discontinuities). Unlike the methods described above, results obtained through Goldstein's method are path

independent, resulting in the non-propagation of errors along an unwrapping direction. The algorithm works by first examining the wrapped phase map for residues and designating them as positive or negative charges. The exact procedure is outlined in [46], which adds some extra steps omitted in the original paper. The pixels immediately surrounding each residue are scanned to detect if another residue is nearby, and if detected they are joined to the central residue with a branch cut. Any other residues detected in this region are similarly joined by branch cuts until the charge is neutral. If this does not result in a neutral charge then the regions surrounding the detected residues are scanned and branch cuts applied, and if the charges still do not balance then the scan area is enlarged from 3 x 3 to 5 x 5 pixels and the process is repeated until charges are balanced or the edge of the phase map is reached, whereupon a branch cut is made to the edge, having the effect of balancing the charge due to the inability of the algorithm to cross branch cuts and surround the unbalanced residues. It is a feature of the algorithm that branch cuts are minimised in length and made in clusters around grouped residues, reducing the likelihood of branch cuts crossing and producing regions of phase surrounded by branch cuts, which would result in erroneous results with undefined relationships to the surrounding map in these regions. This algorithm can be implemented in real-time to give a live video of the strain variations. Errors in the unwrapping process can occur when branch cuts are misplaced and do not set out correctly the unwrapping path for the unwrapping integrals to follow. Various researchers have proposed methods using quality maps to aid the algorithm to place appropriate branch cuts. Derauw used them to branch cut between residues with a low quality data path between them [51], while Flynn used quality map regions to act in place of branch cuts and guide the unwrapping [52]. Giglia and Pritt used region growing around bad data to contain residues within and neutralising the charge in the region [46].

The iso-phase unwrapping algorithm [45] used in this work is a method driven by determining a map of the pixels with high reliability in their phase and then by unwrapping the phase at each pixel along a line guided by the reliability map. The line is not continuous, breaking at low reliability pixels, which allows it to unwrap around regions of discontinuities and low quality regions without propagating errors from these regions into areas where the phase is known with more confidence.

3.4.3 Fringe improvement

Noise reduction is often applied to wrapped phase maps by convolution with a filter kernel, *H*, which has been designed to perform a specific operation on the data. These

operations can be smoothing, edge detection, or noise reduction. In the practical work of this chapter, the method used to improve the fringes was the sine-cosine method which is discussed at the end of this section after a review of other filter kernel techniques.

The simplest operation to implement is smoothing of the data by way of mean filtering [53]. An example of a filter kernel is shown below:

$$H_{LP} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad 3\text{-}33$$

This kernel will perform a low-pass filtering operation on the data, smoothing the image by setting the value of each pixel to be the mean of the nine pixels, including itself, in the surrounding region. The visual effect of such a filter is to blur out the detail in the image, in effect reducing noise. This also removes the definition of any edges (phase discontinuities) in the image, which could impact on the effectiveness of the unwrapping algorithms in section 3.4.2 if the magnitudes of the discontinuities are reduced by too great an amount. For example, Itoh's method looks for a discontinuity, $\delta\phi$, between neighbouring pixels of $|\delta\phi| = (0.9 \times 2\pi)$, smoothing the transition over the discontinuity could mask the presence of a phase wrap if that condition is no longer met after smoothing.

An alternative to the mean filter is a median filter [53], which has the effect of reducing noise described as "salt and pepper" noise; randomly distributed high and low pixels. This works by taking the nine pixels in the local region, sorting them into numerical order and then taking the middle value from the list and assigning that to the pixel at the centre of the local region. This filter variant does not blur sharp edges, and so can be applied repeatedly.

An image can be sharpened using a high-pass filter, which is a kernel that enhances the central pixel of the local region when there is a variation in the values of the local pixels, or otherwise sets the central pixel to zero when there is no variation between the local pixels [53]. This is achieved using a kernel that, unlike the mean filter kernel, sums to zero. An example is shown below:

$$H_{HP} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad 3\text{-}34$$

Application of this kernel to an image has the effect of sharpening the image through emphasising the pixels at the centre of high variation regions, giving the impression of highlighting the edges of any detail.

Another method to sharpen an image is to apply an edge detection filter, such as the simple vertical and horizontal kernels shown in equation 3-35 which would emphasise the edges of fringes aligned vertically and horizontally respectively in the image [53].

$$H_{VER} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}; \; H_{HOR} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \qquad 3\text{-}35$$

These styles of kernels applied to emphasise specifically aligned fringes are not robust enough to cope with fringes which may be changing orientation across the image. A more general style of kernel is required for situations where the fringes are not well aligned. The centre-weighted kernels of Sobel [53] can be applied sequentially to emphasise edges in the images which can have arbitrary orientations. Examples of these are shown below:

$$H_{Sobel\,V} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}; \; H_{Sobel\,H} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad \textit{3-36}$$

An effective technique for removing noise from fringe patterns is to filter the image using sine-cosine filtering [54]. A low-pass filter is convolved with the two images that result from taking the sine and cosine of the fringe pattern image, which are then divided to obtain the tangent, restoring the phase pattern but with reduced noise. The advantage of this technique is that the process can be repeated multiple times to reduce noise while maintaining the sharp details of the phase transitions.

3.4.4 Strain calculation

The use of phase extraction methods to obtain the phase of the speckle patterns was presented in section 3.4.1 (an alternative spatial approach to phase measurement is presented in section 4.4). It can be seen from equations 3-6 and 3-7 that a phase map for a given sensitivity vector and image shear will contain contributions to the phase from a mixture of the surface strain components present. In this section, separation of the various strain components from phase maps is discussed.

The various strains in the system are related to combinations of the displacement gradient components that are manifest in the measured phase maps. The full strain field of an object can be represented by the strain tensor, *S*, shown in equation 3-37 [20], where tensile strain, ε's, is given by the diagonal components, and the shear strain, γ's, by the secondary diagonals:

$$S = \begin{bmatrix} \varepsilon_x & \gamma_{xy} & \gamma_{xz} \\ \gamma_{yx} & \varepsilon_y & \gamma_{yz} \\ \gamma_{zx} & \gamma_{zy} & \varepsilon_z \end{bmatrix} \qquad 3\text{-}37$$

Expanding the strain tensor in terms of displacement gradients yields equation 3-38 [55], which allows the determination of quantitative strain in the material being measured and relates shearographic phase to strain.

$$S = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{1}{2}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) & \frac{1}{2}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) \\ \frac{1}{2}\left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\right) & \frac{\partial v}{\partial y} & \frac{1}{2}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) \\ \frac{1}{2}\left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}\right) & \frac{1}{2}\left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z}\right) & \frac{\partial w}{\partial z} \end{bmatrix} \qquad 3\text{-}38$$

This expression shows the components of strain to which shearography is sensitive, namely the strain components, ε_x, ε_y, γ_{xy}, γ_{yx}, that are described only by the surface displacement gradients. It also includes terms for the bulk strain in the material, ε_z, γ_{xz}, γ_{zx}, γ_{zy}, γ_{yz}, which contain partial derivatives of u, v, and w with respect to z, to which shearography systems are generally insensitive, and so these cannot normally be determined. It is worth noting that the sample being measured would likely have to be mostly transparent in order to detect the changes from within the bulk of the material, and that as shearography requires non-specular reflections, measurements from transparent objects can be problematic, though Catalan [56] obtained strain information from lightly scored transparent objects, though not in volume as yet.

Due to the mixing of components in a single phase map, a system with a single illumination source and viewpoint will produce only a qualitative impression of the observed strain loading event. To obtain quantitative data, more measurement directions are required to allow the separation of the terms in the phase equations. This could take the form of either additional illumination sources, or additional observation positions. Taking measurements from three viewpoints, for instance, would allow three phase maps to be obtained, though each map would contain contributions to the phase from the three surface strain components, the partial derivatives with respect to u, v and w in the direction of the applied image shear x or y. These could be solved for by treating the three phase maps as a series of three simultaneous equations. A full set of shearographic phase maps required to obtain a full three-dimensional representation of the surface strain of an object placed under load can be represented in the matrix forms:

$$\begin{bmatrix}\Delta\phi_{x1}\\ \Delta\phi_{x2}\\ \Delta\phi_{x3}\end{bmatrix}=\frac{2\pi\delta x}{\lambda}\begin{bmatrix}k_{x1} & k_{y1} & k_{z1}\\ k_{x2} & k_{y2} & k_{z2}\\ k_{x3} & k_{y3} & k_{z3}\end{bmatrix}\begin{bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial v}{\partial x}\\ \frac{\partial w}{\partial x}\end{bmatrix}$$

$$\begin{bmatrix}\Delta\phi_{y1}\\ \Delta\phi_{y2}\\ \Delta\phi_{y3}\end{bmatrix}=\frac{2\pi\delta y}{\lambda}\begin{bmatrix}k_{x1} & k_{y1} & k_{z1}\\ k_{x2} & k_{y2} & k_{z2}\\ k_{x3} & k_{y3} & k_{z3}\end{bmatrix}\begin{bmatrix}\frac{\partial u}{\partial y}\\ \frac{\partial v}{\partial y}\\ \frac{\partial w}{\partial y}\end{bmatrix}$$

3-39

To extract the displacement gradient components, the equations of 3-39 are rearranged to the following:

$$\begin{bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial v}{\partial x}\\ \frac{\partial w}{\partial x}\end{bmatrix}=\frac{\lambda}{2\pi\delta x}\begin{bmatrix}k_{x1} & k_{y1} & k_{z1}\\ k_{x2} & k_{y2} & k_{z2}\\ k_{x3} & k_{y3} & k_{z3}\end{bmatrix}^{-1}\begin{bmatrix}\Delta\phi_{x1}\\ \Delta\phi_{x2}\\ \Delta\phi_{x3}\end{bmatrix}$$

$$\begin{bmatrix}\frac{\partial u}{\partial y}\\ \frac{\partial v}{\partial y}\\ \frac{\partial w}{\partial y}\end{bmatrix}=\frac{\lambda}{2\pi\delta y}\begin{bmatrix}k_{x1} & k_{y1} & k_{z1}\\ k_{x2} & k_{y2} & k_{z2}\\ k_{x3} & k_{y3} & k_{z3}\end{bmatrix}^{-1}\begin{bmatrix}\Delta\phi_{y1}\\ \Delta\phi_{y2}\\ \Delta\phi_{y3}\end{bmatrix}$$

3-40

This can be extended to include four channels, which has been shown to produce a decrease in the error of around 33% in comparison to using three channels, varying by component. The number of channels may be scaled to any number in order to further reduce errors in the data, though it was shown that increasing the number of channels will also increase the risk of errors propagating from failures in phase unwrapping when combining the results, and that the reduction in noise tends towards a plateau with increasing number of channels, so giving diminishing returns. However, as additional channels may be used to reduce errors in shearography measurements, FDM may be employed to add a channel to a given system for use in this way.

To ease complexity in the calculations, the channels may be arranged to cancel out some contributions to the phase change map. Adjusting the sensitivity vector, i.e. by the alignment of the illumination and observation points, to lie in the *xz*- or *yz*-plane will cause one of the in-plane terms to reduce to zero in equations 3-10 and 3-11 for that channel. This is shown in section 3.3.1, with the in-plane shearography configuration system. While it is possible to reduce one or other of the in-plane terms

to zero, the out of plane term will never vanish due to the $1+\cos\theta$ dependency. This means that the phase maps from in-plane configurations will always contain a mixture of in-plane and out-of-plane surface displacement gradients. Equation 3-13 shows a possible in plane shearography phase change equation for one channel, sensitive to the tensile strain in *x*, and the out-of-plane surface displacement gradient in *z*. By taking two phase change maps from equal and opposite angles about the normal to the object surface in the centre of the field of view, the two coefficients are separable by simple subtraction and addition. The phase change maps obtained from the two channels are represented thus [14]:

$$\Delta\phi_{+} = \frac{2\pi\delta x}{\lambda}\left[\frac{\partial u}{\partial x}k_x + \frac{\partial w}{\partial x}k_z\right] \qquad 3\text{-}41$$

$$\Delta\phi_{-} = \frac{2\pi\delta x}{\lambda}\left[-\frac{\partial u}{\partial x}k_x + \frac{\partial w}{\partial x}k_z\right] \qquad 3\text{-}42$$

Subtracting equations 3-41 and 3-42 yields the result for the in-plane displacement gradient component (shown in equation 3-15), while summing them will yield that of the out-of-plane displacement gradient component (shown in equation 3-16). Thus it can be seen that with the correct choice of shearing direction and relative placement of the illumination and observation axes into a plane, the tensile strain aligned in that plane on the surface of an object can be measured. Other surface displacement gradient terms can be obtained through the same method, dependent on shear direction and the orientation of the plane containing the illumination sources.

3.5 Multiplexing shearography channels using FDM

FDM allows the simultaneous capture of data from multiple channels on a single imaging sensor at full spatial resolution. It can be applied to shearography to multiplex the light captured from more than one illumination channel, allowing the multi-component measurement of strain.

The experimental work presented in this chapter was performed on a two-component, in-plane shearography system, multiplexing the light from the two channels onto a single camera through an image shearing interferometer. The channels in such a system have a common image shear direction and magnitude. The technique uses multiple intensity-modulated laser beams and a single shearing interferometer to provide multiple measurement channels. The resulting speckle patterns are therefore received simultaneously at the imaging sensor and their evolution over time is captured in a sequence of images over a number of modulation cycles.

The time-series of a given pixel, $I(t)$, will vary in intensity over time as a function of the sum of all of the modulation frequencies present.

$$I(t) = \sum_{\mathrm{n}} I_n(t) + C \qquad 3\text{-}43$$

Where $I_n(t)$ is the modulation signal of channel n, and C is the background intensity.

To de-multiplex the channels, the power spectrum is calculated via the discrete Fourier transform (DFT) on a pixel-by-pixel basis. The peaks corresponding to the different channel modulation frequencies are separated using window functions and the intensity is evaluated. The root-mean-squared (RMS) intensity, I_{RMS}, of a channel's signal can be calculated in the frequency domain using Parseval's theorem:

$$I_{RMS} = \sigma_{I_n} = \sqrt{\frac{\sum |I_n(t)|^2}{N}} = \sqrt{\frac{\sum |\mathcal{F}\{I_n(t)\}|^2}{N^2}} \qquad 3\text{-}44$$

Where $\mathcal{F}\{I_n(t)\}$ is the windowed DFT spectrum for the signal, $I_n(t)$, and N is the number of samples. As the DC component of the signal is removed when windowing in the frequency domain, the RMS is equivalent to the standard deviation of the signal, σ_{I_n}. This is repeated for each channel present and the values placed into separate image arrays. These images can then be considered equivalent to speckle patterns captured with only a single illumination channel present at any one time. The technique is summarised in Figure 3-8, which illustrates the steps of the demultiplexing process.

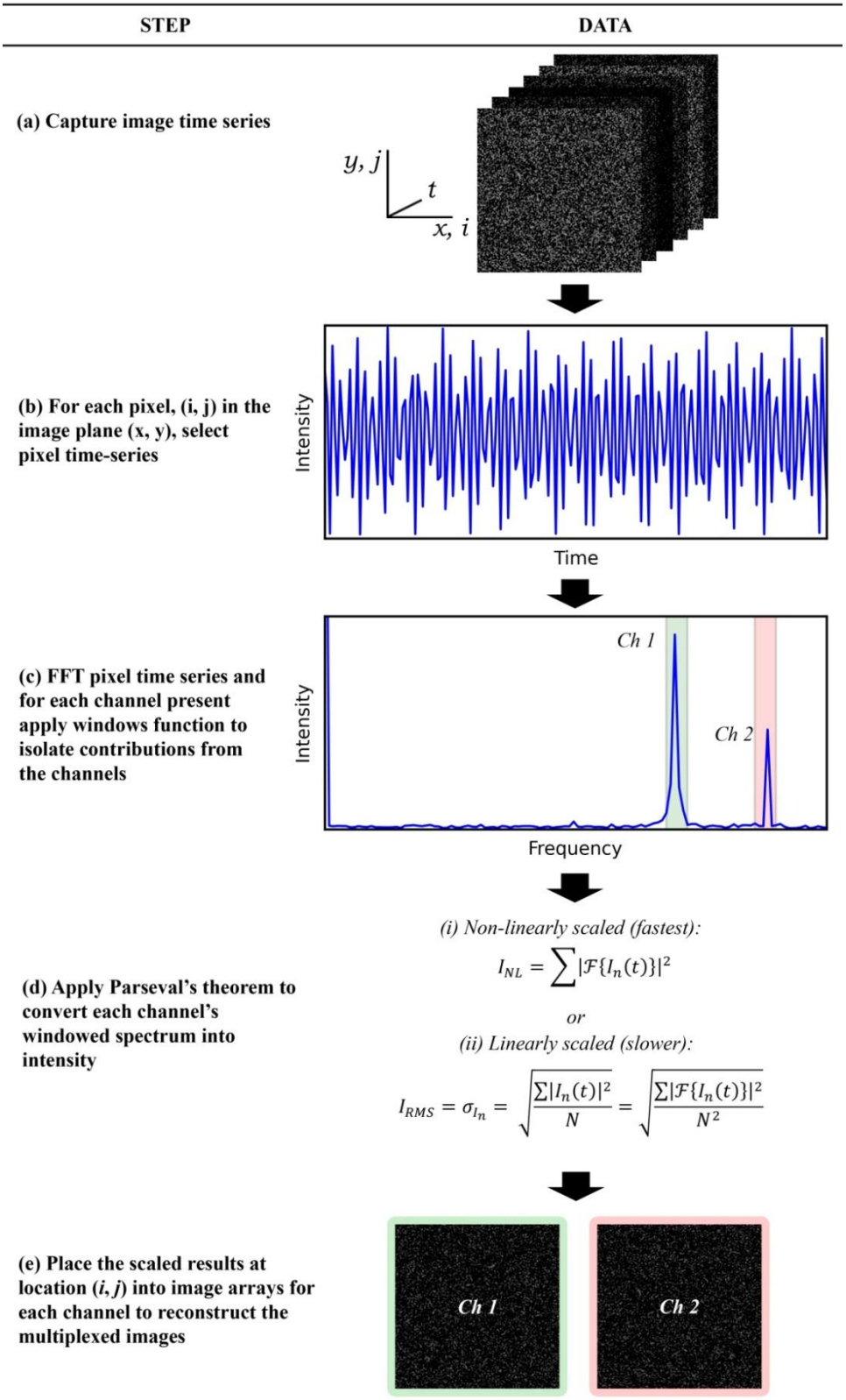


Figure 3-8: Steps to demultiplex images from an image time-series. The example shown is for two channel demultiplexing. Reproduced from [9].

As mentioned in section 2.2, it is not always necessary to calculate the peak-to-peak amplitudes and, instead, the sum of the windowed power spectrum can be used to speed de-multiplexing by using the method shown at (d) part (ii) in Figure 3-8 to obtain the non-linearly scaled intensity, I_{NL}. Care should be taken when using the non-linear intensity with some phase stepping algorithms adversely affected by this [4]. Another point to note is that the modulation frequencies should be chosen so that they meet the Nyquist condition and have resolvable peaks in the power spectrum.

3.6 Experimental application

3.6.1 Introduction to the experiment

A two-component, in-plane shearography system was configured to demonstrate the use of FDM in a practical shearography system. The two in-plane measurement channels were multiplexed on a single imaging sensor, using FDM, and they were used to measure strain on the surface of a test object that deformed predominantly in the plane of that surface. The results were compared to shearography measurements of the same deformation, captured using conventional time-division multiplexing methods.

Here, the experimental layout is described, followed by the procedure used to capture the images, before the results are presented and discussed.

3.6.2 Experimental configuration

The experimental configuration is shown in Figure 3-9. Two illumination channels were created by splitting equally the output from the laser (Coherent DPSS 532 – 300), operating with an output power of 200 mW at a wavelength of 532 nm, into two illumination beams that had the same polarisation state. The beams were expanded to illuminate the test object's surface and arranged to illuminate it from equal but opposite angles in the x-z plane.

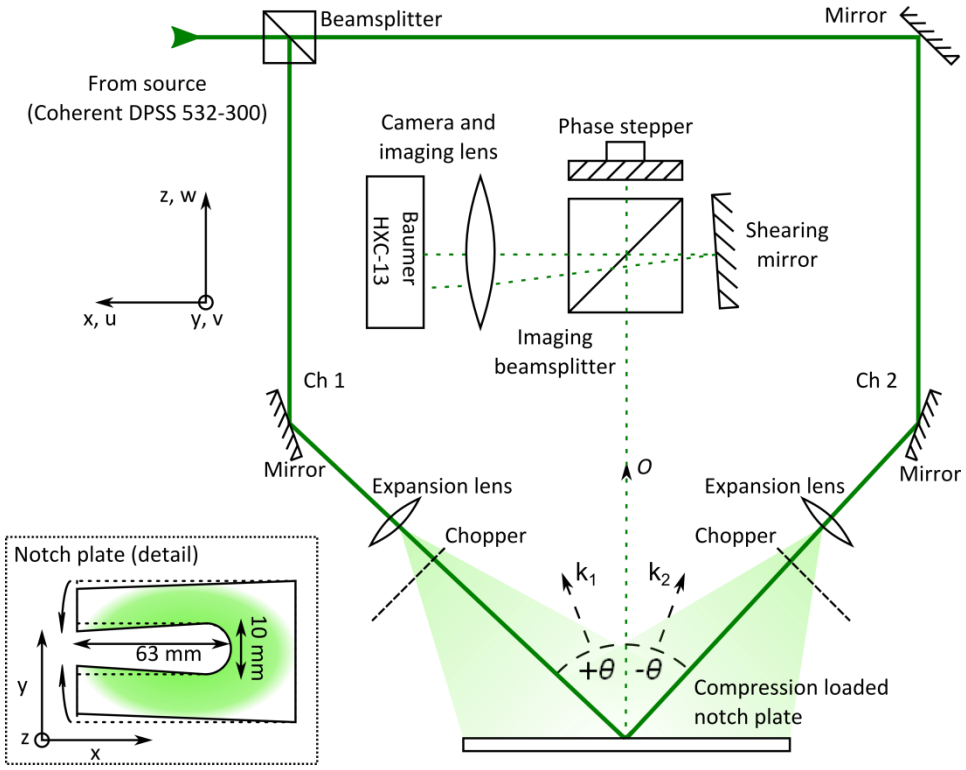


Figure 3-9: In-plane shearography configuration, employing mechanical beam choppers to facilitate the frequency-division-multiplexing of the two channels. Angles ±θ were ±45°, making k_1= 22.5° and k_2 = -22.5° from the observation vector. Inset: The notched-plate test-object as viewed by the system; a 40 mm² area was viewed around the fixed end of the notch. Loading involves compressing the two distal ends together, as shown, using a screw.

This provided sensitivity vectors, k_1 and k_2, of equal magnitude and allowed the separation of the in-plane and out-of-plane components of surface displacement gradient, as per equations 3-15 and 3-16. The illumination intensity on the object's surface was adjusted by varying the divergence of the beams from each channel by selection of appropriate lenses to ensure that the camera did not saturate when both channels were incident on the surface, and also to ensure that both illumination channels had similar intensities. Distinct periodic intensity modulations were applied to the two channels by placement of beam choppers near the focus of the microscope objective lenses used in expanding the beams. These square wave modulations should theoretically produce a comb of frequencies in the power spectrum, consisting of a peak at the fundamental modulation rate and then peaks of successively decreasing height at every odd harmonic. However, in practice, power was also found to have leaked into the even harmonics. It was possible to avoid the emergence of harmonic peaks by modulating at frequencies with harmonics that lay beyond the Nyquist frequency, which caused the harmonic peaks to appear at the other end of the

spectrum to where the channels were located. Modulation frequencies were selected to avoid the fundamental component from one modulation frequency overlapping with harmonic content from the other modulation frequency, and to allow a high number of modulation cycles to be sampled while still fulfilling the Nyquist condition. Thus they were set at 140 Hz and 175 Hz for channels 1 and 2 respectively, as shown in Figure 3-10.

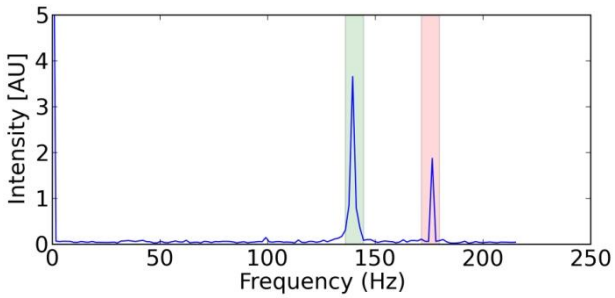


Figure 3-10: The mean power spectrum of an image bank containing two peaks corresponding to the frequencies of the modulation applied to the two channels. The truncated peak at 0 Hz, corresponding to the background light, reached an intensity of 11.5 AU. The shaded regions show the extent of the two window functions used in discriminating channels during de-multiplexing. The peaks had different heights due to a difference in intensities between the two illuminating beams.

A Michelson interferometer was used to provide a lateral image shear of 3 mm, measured against an imaged millimetre scale on the object surface, applied along the y-axis by tilting the mirror of one arm of the interferometer. The mirror of the other arm was mounted on a piezo actuator to provide longitudinal translation of the mirror, allowing use of temporal phase stepping [4]. A telephoto imaging lens arrangement was positioned between the camera (Baumer HXC-13) and the interferometer to image the surface of the object. The object was positioned 0.75 m from the front element of the lens. The camera was configured to use a sub-region of 600 x 600 pixels and a framerate of 430 fps at 8 bit depth, which allowed the capture of 256 frames in the image time-series whilst ensuring sufficient signal levels in each frame.

The test object was a compression-loaded notch in an opaque Perspex block which produced predominantly in-plane deformations, suitable to produce responses detectable with the two-component in-plane shearography system. Loading was achieved using a screw on the end of the notch that was turned and pulled the ends together.

The five-frame temporal phase stepping algorithm shown in equation 3-45 [40] was used to evaluate the phase.

$$\phi = \tan^{-1}\left[\frac{2(I_2 + I_4)}{2I_3 - I_5 - I_1}\right] \qquad 3\text{-}45$$

This algorithm was selected due to its tolerance towards many systematic detector errors [4]. The experimental procedure involved the capture of five time-series, one at each of the five phase steps, the subsequent application of a load to the object and then the capture of another five phase-stepped time-series. An additional image was captured before each phase-step time-series to obtain time-division multiplexed (TDM) data. Doing so allowed for the direct comparison of the performances of TDM and FDM approaches under the same loading conditions.

3.6.3 Procedure

The time-series were processed as described above. The peaks in the power spectrum at the fundamental frequencies were selected using bandpass filters centred on the fundamental frequencies, with a width of 10 Hz to ensure capture of the entire peak. The reconstructed speckle patterns were then converted to a wrapped phase map for the loaded state via the phase stepping algorithm, filtered with five passes of a sine-cosine filter [54] and unwrapped using the 'Iso-phase unwrapping' algorithm [45]. Maps of the change of phase between the reference and loaded states were then calculated, which can be described in the form of equations 3-46 and 3-47 for the two-channels of an in-plane shearography system.

$$\Delta\phi_1 = \frac{2\pi\delta y}{\lambda}\left[k_x\frac{\partial u}{\partial y} + k_z\frac{\partial w}{\partial y}\right] \qquad 3\text{-}46$$

$$\Delta\phi_2 = \frac{2\pi\delta y}{\lambda}\left[-k_x\frac{\partial u}{\partial y} + k_z\frac{\partial w}{\partial y}\right] \qquad 3\text{-}47$$

The values in these two phase change maps were then processed by subtraction to reveal the in-plane strain distribution (equation 3-15), and by summing to reveal the out-of-plane strain distribution (equation 3-16).

The time-division multiplexed images were processed by passing them directly to the phase stepping algorithm, and from there following the same processing steps as used for obtaining the two displacement gradient components with the FDM data.

3.6.4 Results

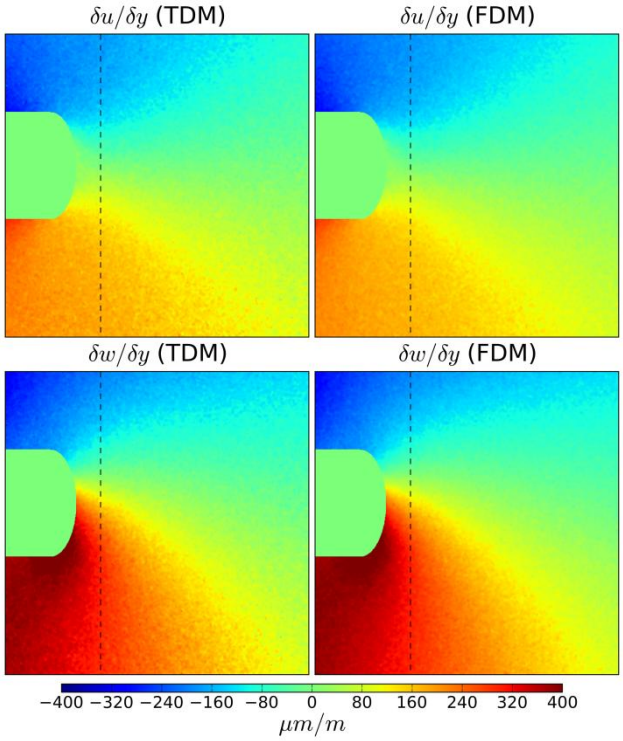


Figure 3-11: The in-plane (top) and out-of-plane (bottom) displacement gradient components of a 41 mm by 41 mm area around a compression loaded notch in a Perspex plate. The left column shows results obtained using TDM and the right column shows those obtained using FDM. The dashed lines indicate the location of the line plots in Figure 3-12.

The in-plane and out-of-plane displacement gradient components on the notch plate are shown in Figure 3-11 for both TDM and FDM. The strain maps for both in-plane and out-of-plane strains are qualitatively similar for the TDM and FDM cases and a quantitative comparison showed that the FDM results agreed to within 1.5% of the TDM method across the field of view. An example vertical cross section of the data shown in Figure 3-11 has been plotted in Figure 3-12, showing the components of strain from beside the notch region. Both components of strain and both multiplexing techniques represented in Figure 3-11 are plotted, and show the agreement between the TDM and FDM results.

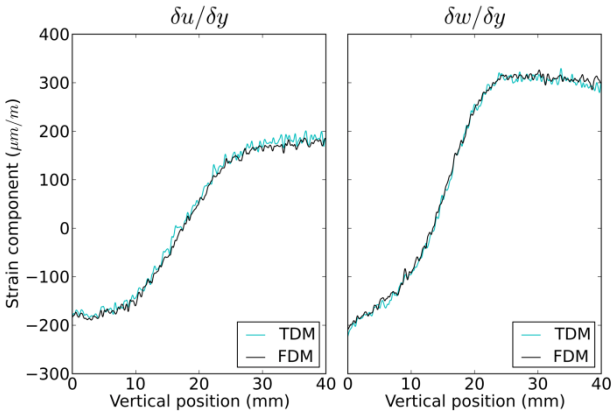


Figure 3-12: A vertical profile taken from the displacement gradient maps shown in Figure 3-10, near the apex of the notch. The left plot shows the profiles for the in-plane component and the right plot is the same for the out-of-plane component.

The results demonstrated that the FDM technique can be applied successfully to a multi-component shearography system and that the results are comparable to the established TDM technique.

3.7 Future work

The system demonstrated utilised two channels multiplexed to give two components of strain. However, the number of channels can be scaled to allow the measurement of more components. The three dimensional strain components for a given shear direction may be measured by the addition of another channel. The three channels would be multiplexed using FDM to enable their simultaneous capture.

The shear directions could be multiplexed in an FDM system, as well, using a combination of FDM and wavelength division multiplexing or polarisation division multiplexing. A scheme for utilising FDM with WDM is shown in Figure 3-13. In this, two wavelengths are emitted at each of the illumination channel positions, giving six FDM channels, C1 to C6, enabling full three-dimensional measurements, as C1, C3, and C5 produce the *x*-sheared components, while C2, C4, and C6 give the *y*-sheared components. Wavelength discrimination inside the interferometer would be achieved by using dichroic beamsplitters designed for the separation of wavelengths. The wavelengths need not be closely separated, but might be, for example, green and red coloured to reduce the tolerances on the optics, provided the sensitivity differences of the two wavelength channels were taken into consideration. All the six channels would be imaged on a single camera. It is worth noting that as the number of channels

increases, the intensity in each is decreased, which would increase the influence of shot noise on the measurements.

It would be interesting to explore the measurement of dynamic systems using FDM shearography. It is conceivable that the modulations applied to the channels may be chosen to enable the freezing of any vibrations on a component and measure the strains during any phase of a vibration cycle without the need for higher powered pulsed sources that have been demonstrated as strobe illumination sources.

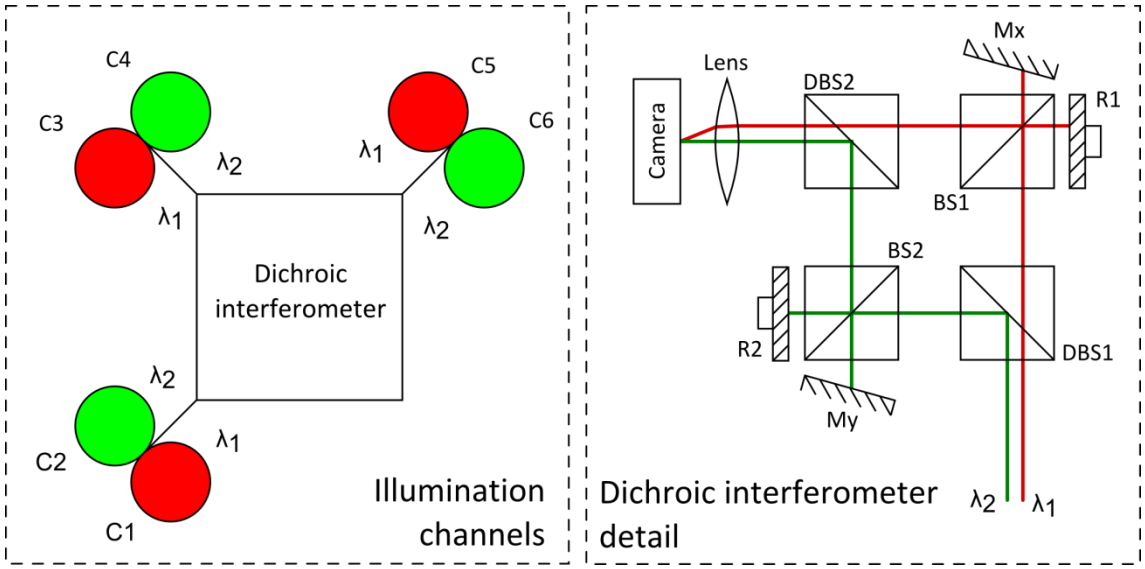


Figure 3-13: Dichroic, shear-multiplexing interferometer and illumination scheme. λ_1 and λ_2: Different wavelength channels; C1-C6: FDM channels; DBS1 and DBS2: Wavelength discriminating beamsplitters for separating and recombining wavelength channels respectively; BS1 and BS2: Beamsplitters; R1 and R2: Phase stepping mirrors; Mx and My: Image shearing mirrors for the x- and y-directions respectively. At each illumination channel there would be two wavelengths emitted, which would be transmitted to the object under test and imaged through the image-shearing dichroic interferometer. One wavelength is transmitted at the first beamsplitter DBS1 to the x-shearing optics and the other is reflected to the y-shearing optics. These two paths are recombined at DBS2 and imaged onto the camera. Channels C1 to C6 each have different intensity modulation rates applied to them to allow the six channels to be multiplexed to the single camera by FDM.

3.8 Conclusions

The development of a new method to multiplex the channels in a shearography system onto a single camera for simultaneous capture has been presented. The FDM method allows the multiplexing of the channels using the full image sensor resolution simultaneously for each channel, with low cross-talk between channels and inherent noise suppression. The relative advantages of the FDM technique compare to other multiplexing techniques are summarised in Table 3-1. Measurements undertaken using the FDM technique have been shown to be comparable to the established TDM technique, agreeing to within 1.5%. Cross-talk between the multiplexed speckle images was shown to be less than 2% of the power in the originating channel, and camera noise at the de-multiplexing frequencies was reduced by a factor in excess of an order of magnitude compared to that of a single-frame.

Table 3-1: Comparison of multiplexed shearography methods (TDM: Time division multiplexed; WDM Wavelength division multiplexed; PDM: Polarisation division multiplexed; SM: Spatially multiplexed; FDM: Frequency division multiplexed). Italic entries are considered unique to the FDM technique.

Method	Pros	Cons
TDM	Use of full image sensor. No cross-talk.	Sequential image capture, therefore only suitable for quasi-dc measurements.
WDM	Use of full image sensor. Simultaneous image capture.	Different wavelength sources required. Cross-talk from wavelength selection.
PDM	Use of full image sensor. Simultaneous image capture.	Cross-talk from polarisation optics. Not suitable on depolarising surfaces.
SM	Simultaneous image capture. No cross-talk.	Uses sub-regions of image sensor or multiple sensors. Requires precise inter-image alignment.
FDM	Use of full image sensor. Simultaneous image capture. *Inherent noise suppression.* *Simple implementation by beam-chopping.* Low cross-talk.	Image time-series required. Dynamic range reduced by increasing number of channels.

It is expected that the multiplexing capacity can be increased to allow for the multiplexing of additional channels, and for multiplexing orthogonal shear directions. This would allow for the simultaneous capture of all of the required information necessary to calculate complete surface strain.

4 Application of FDM to interferometric filter planar Doppler velocimetry

4.1 Introduction

Planar Doppler velocimetry (PDV), also known as Doppler global velocimetry (DGV) or Doppler picture velocimetry (DPV), is a full-field, all-optical method for the measurement of the velocity components in a section of a fluid flow defined by the illumination plane produced from a thin sheet of light. A PDV system generally consists of a number of components; a laser, illumination optics to shape and deliver the light sheet to the flow, and an imaging head which converts Doppler shifts into intensity changes via a spectral filter for detection by a camera. The exact configuration of these components and the methods used can vary significantly and examples will be discussed in this chapter. The technique can have applications in aerospace and turbine design, facilitating the direct optical measurement of fluid flow field without the need to raster scan the measurement plane.

In this chapter, PDV is discussed briefly, including its background and example implementations, and a minor review of the phase and image processing techniques used in the practical work is provided, before a description of the practical demonstration of FDM applied to PDV.

4.2 PDV theory

4.2.1 Principles

In its most basic form, a PDV system contains an illumination source, beam shaping optics to direct the light into a flowing fluid containing a scattering material and an imaging head which contains a transducer to encode the Doppler shift of the scattered light into intensity variations that can be detected by a digital camera. The transducers can be either molecular absorption based (M-PDV) or interferometric (I-PDV).

PDV systems use beam shaping optics to form a laser beam into what is termed a light sheet, which is a very tall and narrow beam. This is used to illuminate a plane through the flow being measured. The sensitivity vector of the system is determined as the bisector of the angle between the direction of propagation of the light in the sheet and the viewing direction of the imaging head. This determines the velocity vector to which the system is sensitive.

The flow contains particles that are entrained in the path of the flow and which are small enough to be supported without slipping and without altering the flow acoustically or thermally [57]. These particles, typically with diameters on the order of the wavelength of light being used to illuminate the flow [58], scatter the incident light from the light sheet as they are carried by the flowing fluid. PDV uses generally Mie scattering of the light, which comes from particles of diameter on the order of one tenth the wavelength of light or larger interrupting the propagation of the light and scattering it from the particle centre. The scattering is not uniform, but has a lobed structure with high and low intensity at various angles relative to the initial propagation direction of the light, though it is a predominantly forward scattering process [58].

The light scattered from the particles is Doppler shifted by an amount, $\Delta\nu$, related to the average velocity of the particles, V, to the projection of the velocity components onto the sensitivity vector and to the optical frequency of the un-shifted light, ν_0. The sensitivity vector dictates the proportions of each of the velocity components that the system is sensitive to for a given direction of observation, $\hat{\imath}$, and illumination, $\hat{o}$, given as the projection of these components onto the sensitivity vector. The sensitivity vector is the vector bisecting the angle between the $\hat{\imath}$ and $\hat{o}$ vectors. Equation 4-1 [59] shows the relationship between the velocity of the illuminated particles in the flow and the Doppler shift.

$$\Delta\nu = \frac{\nu_0(\hat{\imath} - \hat{o}) \cdot \bar{V}}{c} \qquad 4\text{-}1$$

For most flows, the Doppler shift is typically a small fraction of the illumination frequency, on the order of a few tens to a few hundreds of megahertz. A means needs to be employed to convert this small shift in frequency into a more easily measured quantity, and this is the reason for the use of an optical filter in the imaging head prior to the camera.

The filter is most commonly of two types – either an interference filter or a molecular filter. The practical work presented in this chapter was conducted using an interference filter. The following sections describe the frequency to intensity transducing techniques of the two types of imaging heads.

4.2.2 Interferometric filter PDV (I-PDV)

The Doppler shift described by equation 4-1 can be converted into an intensity change, and therefore made detectable with a digital camera, by using a path length imbalanced interferometer. An interferometer, such as a Michelson [60] or Mach-

Zender [61], would be placed so that the light scattered from the flow would have to pass through the interferometer before reaching the camera. The image recorded by the camera would then consist of the resulting interference pattern overlaid on the image of the flow. Changes in optical frequency will change the phase between the paths of the interferometer, as shown in equation 4-2, where $\Delta\phi$ is the phase change induced, Δl is the optical path length difference between the two arms of the interferometer, c is the speed of light, and $\Delta\nu$ is the Doppler shift.

$$\Delta\phi = \frac{2\pi\Delta l}{c}\Delta\nu \qquad 4\text{-}2$$

Measurements of the phase of light exiting the interferometer can therefore be used to determine the Doppler shift and from there the velocity can be calculated by applying equation 4-1.

The first step in the measurement of the velocity is to make a reference by measuring the phase of the output from the interferometer with no Doppler shift imposed on the input light. Phase measurement is discussed in section 4.4. This can be done either before or after taking a measurement of the phase change due to the velocity of the flow, or simultaneous with the velocity measurement through some form of multiplexing approach, both of which are discussed in section 4.3. The reference phase can be obtained from a slow moving or stationary flow medium, or from a wavefront produced by scattering from a screen. A reference state is required in order to determine the magnitude of the phase shift imposed by the Doppler frequency shift when the flow velocity is measured. To determine the phase of the Doppler shifted signal, the light scattered from the particles in the moving flow is imaged through the interferometer and the chosen phase measurement technique is again applied. The reference and signal light phases are then subtracted to reveal the phase change, $\Delta\phi$, which is related to Doppler shift by equation 4-2, and which is then used to calculate the velocity as per equation 4-1.

The measurement range of the interferometer is defined by the free spectral range (FSR), and this is the maximum Doppler shift that can be measured before the phase becomes wrapped. The FSR is given by:

$$FSR = \frac{c}{\Delta l} \qquad 4\text{-}3$$

To facilitate the measurement of the three orthogonal components of the flow velocity, an interferometer would be required for every imaging head in a multiple observation vector scheme, or a single interferometer and imaging head could be used with multiple illumination vectors. The sensitivity of an I-PDV system to Doppler shift is

controlled by the path-length imbalance of the interferometer, as per equation 4-2; the greater the imbalance, the greater the sensitivity to Doppler shifts, which allows I-PDV to accommodate both high and low velocity flows through tuning the resolution, as low resolutions allow for a higher velocity range. This also affects the measurable unambiguous velocity. Changes in the interferometer, for example, alterations of the path length imbalance caused by thermal expansion of components in paths or air currents, can influence the measured Doppler shifts, though the interferometer can be actively stabilised or isolated to help prevent such drifts.

4.2.3 Molecular filter PDV (M-PDV)

Molecular absorption filters are used to transduce the Doppler shift into an intensity variation that can be measured on a conventional digital camera in M-PDV. The molecular filters take the form of an absorptive gas contained in a gas cell through which the light sheet is imaged. The absorption features in the gas should be within the tuneable range of the laser, be deep and sharp, and, for some approaches, there should be a region to the side of the feature with little variation in absorption to allow the recoding of a reference signal. The absorption features of a gas in a gas cell have fixed wavelengths, though the minimum level of absorption can be increased using Doppler broadening to absorb more light by increasing the partial pressure of the gas, so reducing the gradient on the feature edges. Feature broadening can also be achieved through addition of a non-absorbing gas species to the cell, causing the width of the absorption features to vary dramatically with pressure through Lorentz broadening, allowing the tuning of feature widths to span a desired range to accommodate the expected velocity induced Doppler shifts [58] [59].

The absorptive gas most usually chosen is the diatomic molecular form of iodine, I_2, due to the presence of number of sharp absorption features in the 532 nm region. This gas is the subject of a comprehensive model of absorption developed by Forkey [62]. Other gases are also used, such as rubidium, caesium, potassium, lead and mercury [59]. The choice of gas is dependent upon the region of the spectrum in which the absorption features are located, the availability of suitable optical sources, the strength of the absorption features, the local region of the feature (for example, some techniques require a flat plateau beside the absorption feature), and the operating conditions of the gas cell (e.g. high mass gasses require higher operating temperatures to Doppler broaden) [59].

M-PDV is conducted by tuning the laser to sit half-way up one edge of an absorption feature of the thermally stable gas and then directing this light into the flow in a light

sheet as described in section 4.2.1. The light is Doppler shifted by the moving particles entrained in the flow and the light is collected and imaged through the gas cell. The Doppler shift causes the light to have a slightly different position on the absorption feature, and hence the transmission of the cell changes. This transmission change is detectable at the camera as a change in intensity relative to a reference level taken from light scattered from the flow when it was stationary.

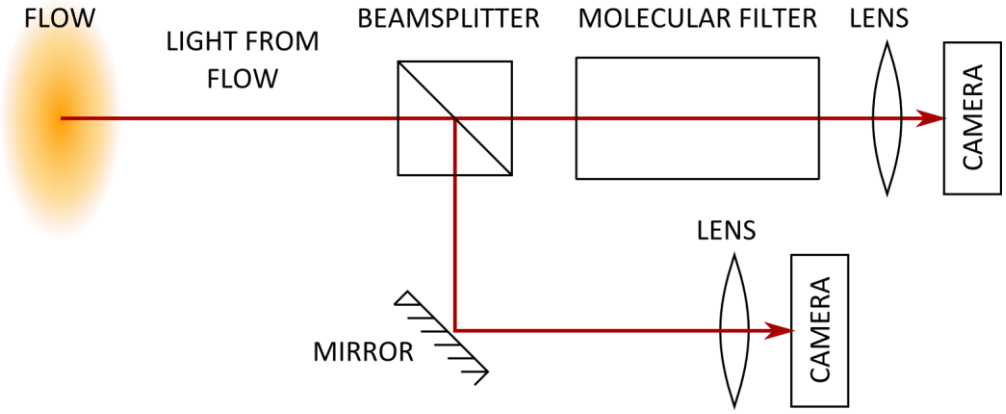


Figure 4-1: Molecular PDV imaging head showing the light path split between two paths, one of which contains the molecular filter.

An imaging head in an M-PDV system records two images, a signal image that has passed through the gas cell, and a reference image that is not filtered by the gas cell used to normalise the signal received (see Figure 4-1), so that variations in light power unrelated to those caused by the Doppler shift do not influence the measured velocity. The unfiltered reference image may be captured by splitting the incoming light into two paths and passing one through the cell (for the Doppler signal) and the other path around the cell (the normalising reference signal). These two paths may be brought together on two halves of the imaging sensor on a single camera, or kept separate and directed onto the full frame of the sensors in two separate cameras. The use of two separate cameras requires that they be aligned to each other to ensure that the images are normalised correctly. This is often achieved through the use of image calibration targets, often grids of points, which are used in image dewarping and interpolation algorithms to register pixels between images. Each component of velocity to be measured requires an imaging head with these characteristics, unless multiple illumination channels are employed [59][58].

Light intensity is used to convert the Doppler shift to a camera readable intensity, but in order to measure the velocity of the flow using equation 4-1, the Doppler shift and not intensity must be used. The Doppler shift is obtained from the optical filter's transfer function, which must be determined from the stable gas cell by scanning the

wavelength of the output from the laser, which is scattered from the flow at rest or from a scattering test object, across the absorption feature and measuring the normalised intensity for each view. This allows the filter transfer function to be plotted as normalised intensity against wavelength or frequency, such that the Doppler shift can be read from the transfer function.

The main issue with M-PDV is that the gas cell must be kept stable, as the absorption features change with temperature and pressure changes in the cell, which can lead to changes in the transfer function which in turn lead to false measurements of the Doppler shift. The laser and gas must also be matched to one another, and the laser must be tuneable on the order of a one to a few a gigahertz depending on whether the features are broadened by Doppler or Lorentzian processes respectively. Adjustment of the imaging head dynamic range is possible by adjustments of the pressure in the cell or the temperature used to vaporise the gas.

4.3 Configurations

4.3.1 I-PDV configurations

The phase change induced by the Doppler shift can be measured by taking an image of the flow when there is either a zero or low velocity flow [58], and then calculating the phase of the fringes across the image to generate a reference phase image. The flow is then turned on and the phase of the fringes measured again to generate a signal phase image. The signal and reference phase images are then subtracted and the difference between them is obtained, which is equivalent to the Doppler imparted phase shift.

This does not have to be done in a sequential fashion; it is possible to obtain a reference at the same time as the signal state. Seiler [60] demonstrated a path length imbalanced Michelson interferometer based PDV system (see Figure 4-2) that used polarisation selective optics to allow the flow to be imaged onto two cameras through a common interferometer to either one of two planes. One of the planes was located within the flow being measured, while the other plane was located at a diffuser screen that acted as the reference. This system has been developed [63] to use the carrier fringe methods described in section 4.4, which was an improvement on the techniques used prior to Takeda [64], where fringes had to be located by the operator.

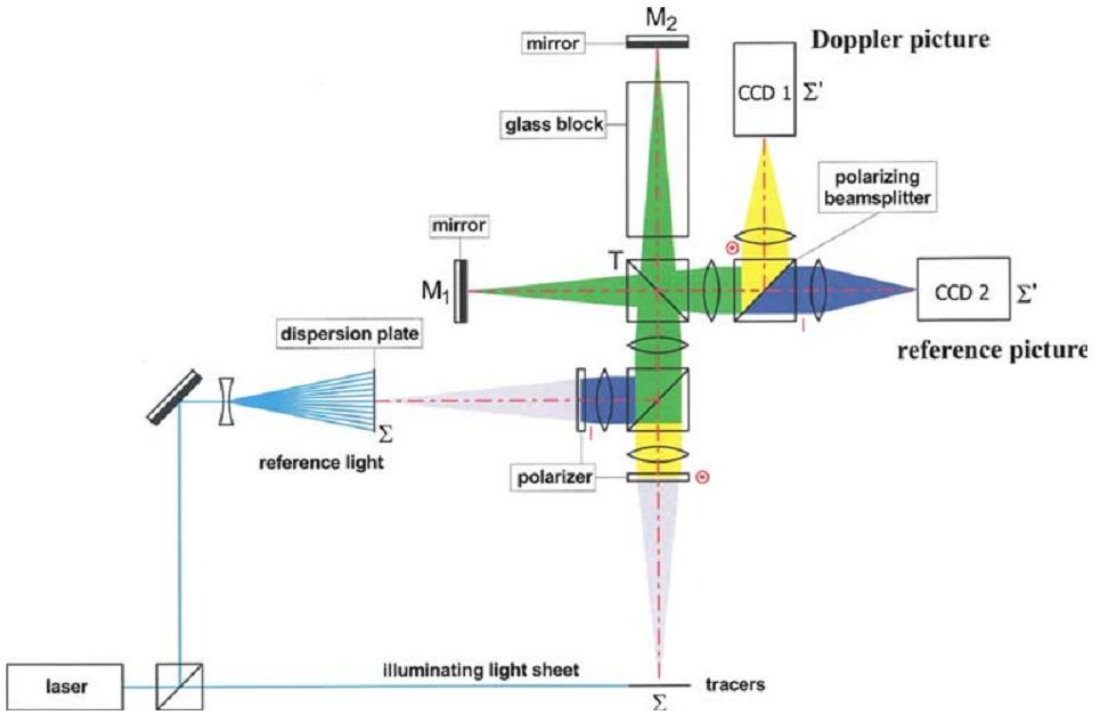


Figure 4-2: The Michelson interferometer and illumination used by Seiler et al (Figure reproduced from [60]). Polarisation is used to select for images between the reference and the illumination image planes. The paths of the light are shown by the colours, blue for the reference, yellow for the scattered light of the flow, and paths common to the two images are in green. Note that there are two CCD cameras used to capture the reference and signal images separately.

Lu [61] described a method which allowed the normalisation of fringes and the reduction of the background light levels to give sharper interference fringes and thus improve measurements. In an MZI filter system this is achieved by using the two output ports of the MZI, which produce two identical fringe patterns, I_1 and I_2, that are in anti-phase to each other. Equation 4-4 is then used to calculate the normalised fringe pattern I_N.

$$I_N = \frac{I_1 - I_2}{I_1 + I_2} \propto V\cos(\Delta\varphi) \qquad 4\text{-}4$$

An advantage of this approach is that it increases the sensitivity of the system as it makes the fringe modulation depth twice as deep due to the subtraction of the two anti-phase fringe patterns, while the division by the sum of these patterns allows normalisation and removal of the background.

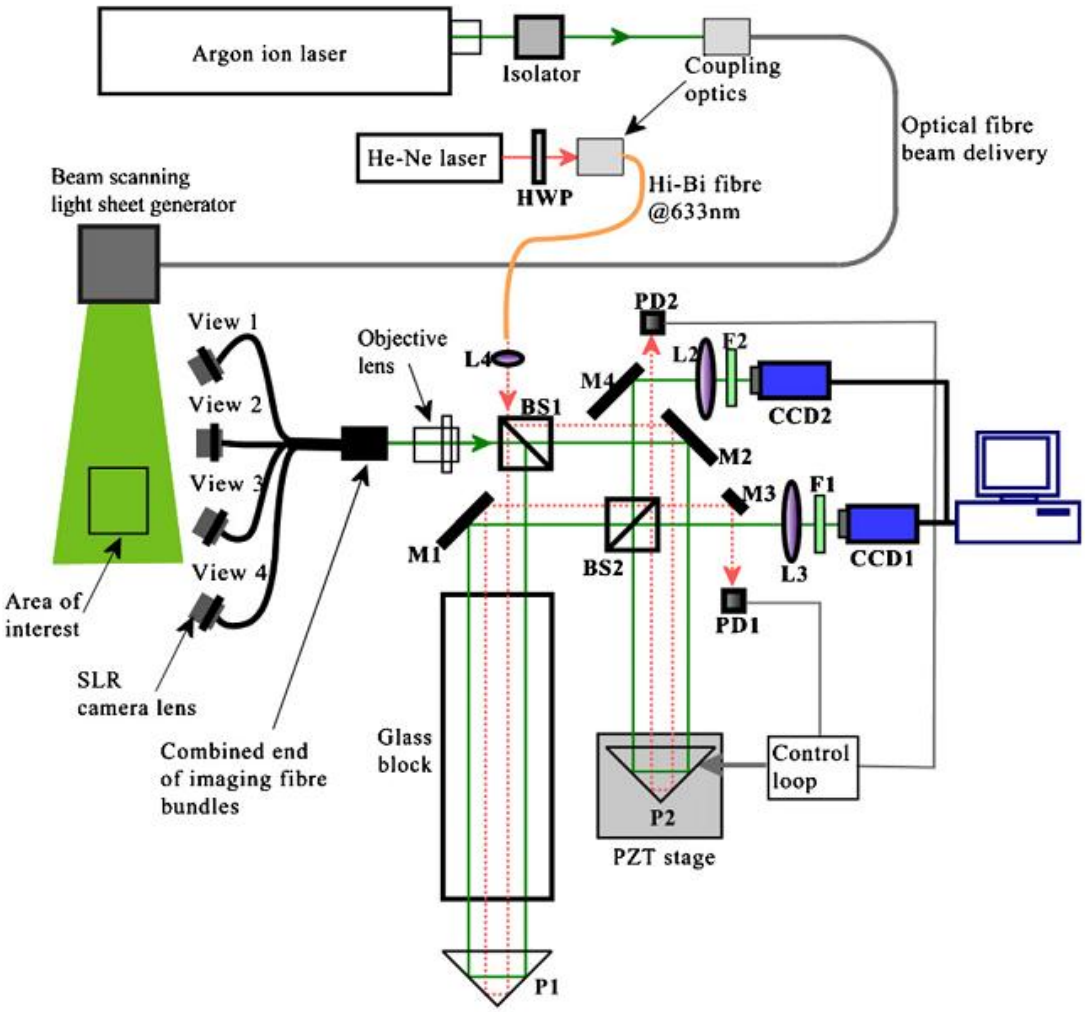


Figure 4-3: The path length imbalanced MZI used by Lu et al (Figure reproduced from [65]). It features the multiplexing of multiple views to a single input port of the MZI and two CCD cameras to perform the fringe normalisation step on the two output ports of the MZI. There is active stabilisation of the interferometer to avoid fringe drifts, using a HeNe laser to monitor for drifts and provide feedback to the translatable prism P2 which can be moved to compensate for these drifts.

The phase was determined using the carrier fringe method in both the systems of Lu et al [61] and Seiler et al [60].

Landolt and Roesgen reported a system [66] with a variable path length imbalance which allowed greater sensitivity to Doppler shift. An iodine vapour cell was placed in one arm of a Michelson interferometer, and the laser illumination source (frequency doubled Nd:YAG) was tuned to a resonant spectral absorption feature in the iodine vapour spectrum. This approach represents a hybrid of the two techniques of molecular-filter and interferometric-filter PDV. The refractive index of the iodine

vapour changed as the Doppler shift moved the laser frequency about the edge of the absorption feature. This, coupled with the optical path length imbalance of the two arms, increased the sensitivity of the Michelson interferometer.

4.3.2 M-PDV configurations

Though not the technique used in the practical work of this chapter, there are some notable implementations of M-PDV that are discussed here for their suitability for the use of FDM multiplexing.

The most common instance of M-PDV imaging configurations is the type shown in Figure 4-4. This method has been employed in the wind tunnels at the National Full Scale Aerodynamics Laboratory at Ames Research Center, and is reported in [67]. The method uses two cameras to capture the reference and the signal light, which adds to the cost and the complexity of the imaging head. An alternative approach to this was demonstrated by Charrett in two-frequency PDV, where the laser source was modulated between two frequencies, one inside and the other outside the iodine absorption feature, allowing a single camera to be used to capture both signal and reference images sequentially [68]. The light sheet in that work was generated not by using a cylindrical lens arrangement to expand the beam, but by scanning galvanometers to create the light sheet in a time averaged way, with three scans of the beam to each image capture.

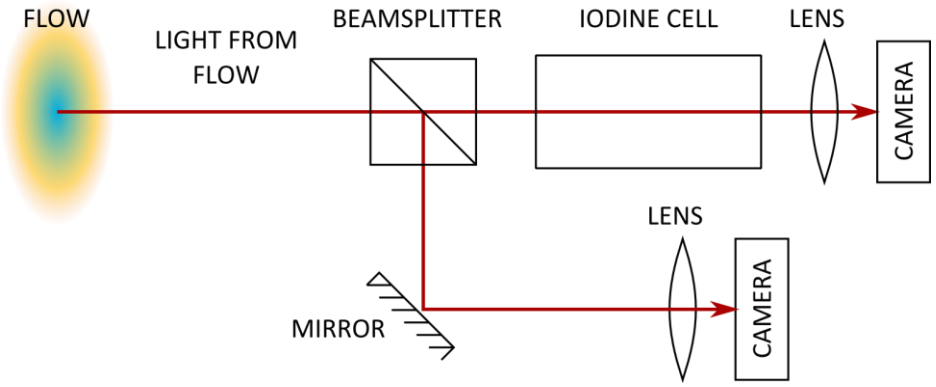


Figure 4-4: A standard M-PDV imaging configuration. This is the method chosen by the National Full Scale Aerodynamics Laboratory at Ames Research Center. It uses two cameras, one for the reference intensity and the other measures the intensity through the iodine vapour cell.

Fischer et al proposed frequency modulation DGV (FM-DGV), which uses a molecular filter without the need for a reference image, by using frequency modulation of the laser source around the absorption feature. This system has been described for a

single-point [33] and for multiple-point measurements [69]). The method used a periodic modulation of the laser centre frequency about some value in the absorption region. This modulation caused the absorption of the light to vary in time as the laser centre frequency moved from the high absorption centre of the absorption feature to the lower absorption regions at the feature edges. As this happened periodically, the intensity measured at a point in space over time also took on a periodic form. The measured signal consisted of a pair of spectral components, revealed when analysed with an FFT, due to the asymmetry of the absorption feature caused by variations in the absorption mechanism and their strengths across the region. The ratio of the two peak intensities could be used to give a measurement quotient, *q(f)*, related non-linearly to the centre frequency of the laser and hence to any Doppler shift imparted by the flow off which the laser light was scattered.

The FM-DGV technique has been further expanded to allow the simultaneous measurement of different velocity components, in effect using the modulation frequency to enable multiplexing[1]. Two laser sources tuned to the same point in an absorption feature are modulated at two different rates, which causes the pairs of peaks from either source to appear separated, allowing the peak-pairs to be sampled from two scattering directions in the same data set. The lasers were positioned perpendicularly at either side of the flow being measured, allowing two velocity components to be measured.

4.4 Spatial phase measurement

4.4.1 Introduction

The phase in the interferometer can be measured by a number of means. Temporal phase measurements are a possibility, provided the flow is stable for the duration of the measurements. This type of measurement has been discussed in more detail in section 3.4.1, where it was applied to the shearographic measurement of surface strain. To avoid repetition it will not be discussed here.

Spatial methods allow the measurement of phase by processing information that is captured in the time taken to produce a single frame. This is commonly achieved from a single image, though there are some variants that obtain the information from multiple images spatially distributed throughout a single frame. This might be more appropriate in cases where, for example, the flow is not stable across the capture period of a number of frames or where it would be otherwise desirable to reduce the measurement duration. Spatial techniques can be divided into two broad areas of

operation: phase stepped methods and carrier fringe methods [64]. This section will concentrate on two commonly applied spatial carrier methods, the Takeda and carrier-fringe methods, and also on the spatially distributed methods in use. Other less common methods will also be highlighted.

The carrier fringe method of section 4.4.3 was applied in the FDM I-PDV practical work described in this chapter due to the robust nature of the algorithm in dealing with spatial distortions in the fringes in comparison to the Takeda method.

4.4.2 Takeda method

The phase of light exiting an interferometer can be determined using a 2D Fourier transform of the image of the interference fringes. In order to achieve this, Takeda [2] demonstrated that isolating the peak in a 2D DFT of an image taken through an interferometer with a linear phase shift across the field of view, which produced straight line fringes over the image, allowed the measurement of the phase of the fringes in the image and hence that of the interferometer.

It can be shown that a two-dimensional interference pattern of linear phase lines, $g(x,y)$, may be represented as:

$$g(x,y) = a(x,y) + c(x,y)e^{2\pi i f_0 x} + c^*(x,y)e^{-2\pi i f_0 x} \qquad 4\text{-}5$$

where

$$c(x,y) = \frac{1}{2}b(x,y)e^{i\phi(x,y)} \qquad 4\text{-}6$$

and $a(x,y)$ is the background light distribution and $b(x,y)$ is the intensity function of the interference pattern. By taking the DFT of equation 4-5, the following result is obtained:

$$G(f_x,f_y) = A(f_x,f_y) + C(f_x - f_a, f_y - f_b) + C^*(f_x + f_a, f_y + f_b) \qquad 4\text{-}7$$

where, respectively, f_x and f_y are the x- and y-components of the frequency, f_a and f_b are the carrier fringe frequency x- and y-components, and G, A, and C are the frequency space representations of g, a, and c. The last two terms of equation 4-5 are those which carry the phase information. Isolating the un-conjugated term with a filter function, $H(f_x - f_a, f_y - f_b)$, translating it to the origin of the frequency space and taking the inverse DFT allows the isolation of the complex term $c(x,y)$. It is then a case of performing a division of the real and imaginary components of this image and taking the arctangent to obtain the wrapped phase map, as shown:

$$\phi(x,y) = \tan^{-1}\frac{Im\big(c(x,y)\big)}{Re\big(c(x,y)\big)} \qquad 4\text{-}8$$

This phase map may need to be unwrapped if the phase varies by more than 2π in the image (phase unwrapping is discussed in section 3.4.2).

An advantage of processing phase information from carrier fringes is that there is no requirement for a moving component inside the interferometer, such as a piezo-driven phase stepper, and so less equipment calibration is required.

4.4.3 Carrier fringe method

The carrier fringe method is a modification of the Takeda method described in section 4.4.2, where the isolated peak in the two-dimensional FFT of the fringe image is not translated to the centre of the DFT image before taking the inverse DFT of that resulting frequency space image, but is left in place. This renders the method less sensitive to any distortions that may be present in the fringes.

Frankowski et al[70] investigated the performance of this algorithm through computer simulation of the determination of phase from linear tilt fringes. The influence of Gaussian noise in a fringe pattern was shown to reduce the accuracy of the phase measurement from greater than $\lambda/10^5$ to around $\lambda/200$ as the noise in the fringes increased from 0% to 5% of the intensity. It could be seen that the sampling of the fringes could affect the outcome of the phase measurement; the accuracy peaked at approximately $\lambda/10^6$ when sampling at three pixels per fringe on a fringe pattern without noise. In a more realistic case of 5% Gaussian noise in the fringes, the accuracy was $\lambda/450$ at three pixels per fringe, reducing to $\lambda/100$ when sampling at fifty pixels per fringe. The influence of the digital level assigned to the intensity incident on a pixel in the camera by the analogue to digital converter was shown to have a significant influence on the phase accuracy, an effect of quantisation error. If a fringe image is incident on a camera and the intensity of the light is sufficient to just reach the saturation point of the pixels at the peaks and the visibility is unity, then the fringes will be divided into exactly 2^{bits} number of digital levels, determined by the bit-depth of the analogue to digital converter in the camera. If the visibility of the fringes is reduced or the fringe intensity is reduced, then the number of levels assigned is reduced, as these are distributed throughout the full range of intensities possible to capture for a given exposure time. Reducing the number of levels used to sample the fringe intensity results in a lower resolution fringe pattern with a stepped intensity profile, which in turn reduces the accuracy of the phase. Errors due to quantisation in

the two-dimensional DFT methods are shown to be low, for example, 6 bit digitised data gives a phase error of $\lambda/10^3$ [64].

4.4.4 Quiroga method for phase extraction

The method devised for fringe normalisation by Quiroga et al. [71] can be used to determine the phase of a wavefront. The flow or object under test is imaged through an interferometer which imposes a fringe pattern on the image. This is captured on a digital camera, and the resulting data is filtered in the Fourier domain using a filter based around a combination of the Heaviside unit step function and the circular high-pass filter. This creates a filter which is in effect a band pass filter designed to break the Hermitian symmetry of the Fourier plane and pass only the positive frequencies, with the exception of the low frequency components in a near-semi-circular sector centred on the 0 Hz peak. This type of filter allows arbitrary fringe patterns to be employed, removing the limitation of the carrier fringe and Takeda methods, which must be linear in order to work.

The filtered image, i_1, is obtained as a convolution of the original image, i_0, and the filter, h,to give the result in equation 4-9.

$$i_1(r) = i_0(r) \otimes h(r) = m(r)e^{i\phi} \qquad 4\text{-}9$$

This result is then processed using equation 4-8 and the phase can be recovered.

The issue with this technique is that the filter function has a directionality associated with it that causes fringes to lose resolution in that direction. This means that in places where the fringes lie parallel to the filter the phase cannot be recovered without increasing error in that region. To remedy this, the technique can be performed slightly differently, incorporating a fringe normalisation step into the calculation to allow the recovery of the lost fringe information. This is described in section 4.5.2 and should be used in preference to this technique without these modifications.

4.4.5 Spatially displaced methods

Other methods that can be used for phase measurement involve displacing phase stepped images across a single image sensor, or across multiple image sensors. These techniques generate phase stepped images in parallel, allowing the capture of all of the images required for a given phase stepping algorithm.

Phase stepped images may be generated by a stepped mirror if the phase gradient across the steps is low, approximating a spatially interleaved set of phase stepped

images with the image resolution across the sensor reduced by a fraction of the number of steps required.

The images may also be phase stepped and distributed across the sensor as whole images, instead of interleaved, using a diffraction grating to impart an angular separation as well as a phase offset to the images on the diffraction orders. This was explored by Francis in [21], though it was found that producing gratings that do both image displacement and phase was difficult and that placing a phase mask after the image displacing grating was a more practical approach. Grating displacement is suggested as another method of introducing the phase shift onto the diffraction orders [64].

4.5 Denoising and data improvement

4.5.1 Fourier domain filtering

It is possible to filter an image in Fourier space to improve the quality of the image. An ideal high pass filter allows the emphasis of structure in an image over the background and slowly varying features. The ideal low pass filter is defined in two-dimensional frequency space by[3]:

$$H(f_x, f_y) = \begin{cases} 0, & \sqrt{f_x^2 + f_y^2} \leq f_c \\ 1, & \sqrt{f_x^2 + f_y^2} > f_c \end{cases} \qquad 4\text{-}10$$

where f_c is the cut-off frequency and f_x and f_y are the x and y components of frequency respectively. Reversing the inequalities will produce the low-pass filter.

Other filter types are possible, depending on the desired effects, such as the edge-emphasis filter of the type discussed in section 3.4.3. These filter kernels are applied in convolution with an image, which is in effect a multiplication of the two-dimensional Fourier transform of the image with that of the kernel. Filters can be combined to create hybrid filters such as the orthogonal band pass filters in Quiroga's method for fringe normalisation.

If a Fourier based method of phase extraction is used during image processing, then a filter can be applied at the same time as the peaks in the 2D DFT are selected.

4.5.2 Quiroga method for fringe normalisation

The method devised for fringe normalisation by Quiroga et al. [71] can be used to determine the phase of a fringe pattern by the method described in section 4.4.4, though the technique as it is described there has some limitations imposed due to the directionality of the filter function.

The modification to the method described here allows the determination of the phase in the region where the fringes are lost due to the directionality of the filter. This is achieved by way of using a pair of orthogonal filters to produce two filtered representations of the fringes and then combining the results. These filters are based around a combination of the Heaviside unit step function and the circular high-pass filter. This creates two filters which are in effect orthogonal band pass filters designed to pass only the positive frequencies in their respective directions with the exclusion of the low frequency components around the 0 Hz peak. The two filters will remove information of features orientated in two different directions, but the information lost from one filter can be recovered from the other during processing, maintaining the features of the image which might otherwise have been blurred out.

This is shown by extending equation 4-9 to create the two representations, i_1 and i_2.

$$\begin{aligned} i_1(r) &= i_0(r) \otimes h_1(r) = m_1(r) e^{i\phi_1} \\ i_2(r) &= i_0(r) \otimes h_2(r) = m_2(r) e^{i\phi_2} \end{aligned} \qquad \textit{4-11}$$

These are then processed using the following function to recover the information from the two representations and create a new representation of the filtered fringe pattern:

$$i'(r) = \cos\big(\phi(r)\big) \approx \frac{m_1(r)\cos\big(\phi_2(r)\big) + m_2(r)\cos\big(\phi_2(r)\big)}{m_1(r) + m_2(r)} \qquad \textit{4-12}$$

The new representation of the filtered fringe pattern is a normalised representation of the fringes, ranging between -1 and 1, which is then processed by equation 4-8.

This method up to equation 4-12 can be applied to any fringe pattern to normalise the fringes prior to processing if desired, as it can improve the quality of the fringes, removing background irradiance and distortions. The method is limited in success to regions where the fringes are of high enough spatial frequency to pass the filter function, though the diameter of the circular high-pass filter on the DC component can be varied to suit the conditions of the experiment.

4.5.3 Image dewarping

Image dewarping was conducted in the practical work of this chapter to correct for perspective distortions in the images of the measurement regions. The algorithm applied was based on the work of Willert [72] and implemented in the Python programming language as part of the "impy" module used during previous work at Cranfield University by Dr T. O. H. Charrett.

The algorithm required a number of control points, locations in the original image with known locations in space, which can be found through some means (in the impy implementation this was algorithmically) and then mapped from the coordinates in the original image to new coordinates in the dewarped image, which considers how the points in three-dimensional space are mapped to a two-dimensional plane without the distortion of perspective.

The first step in applying the dewarping algorithm was to determine the location of the image plane in the experiment and place a calibration target in that plane. The target was chosen to be a regular grid pattern of control points, which could be circles, crosses, etc, that could be searched for using a system of localised search regions. An estimate of the initial positions of three spots was given to the search algorithm; an origin spot, and then the next spot along the row from there and the one on the row above. These approximately defined the search vectors for shifting the local search regions. Inside each local region, the control point shape was searched for by convolution of the local region image to the control point image kernel. Once that position was found, the local region was shifted along the search vector and the process repeated until the entire image had been covered. Figure 4-5(a) shows the algorithm having found the control points on a calibration target image as it is viewed from the camera. The algorithm then took the array of points found and mapped them from the locations on the original image onto a dewarped image, which was an attempt by the algorithm to restore the array back to the form it had as the calibration target.

The algorithm used to map the points between the images was based on a least-squares fit to the array of found points, using the Marquardt-Levenberg approach which is summarised in Willert's paper [72]. Including the least squares step in the dewarping algorithm increases the accuracy of the dewarping over the use of the array of found control points. Figure 4-5(b) shows the result of dewarping the image based on the control points found in part (a) of the same figure.

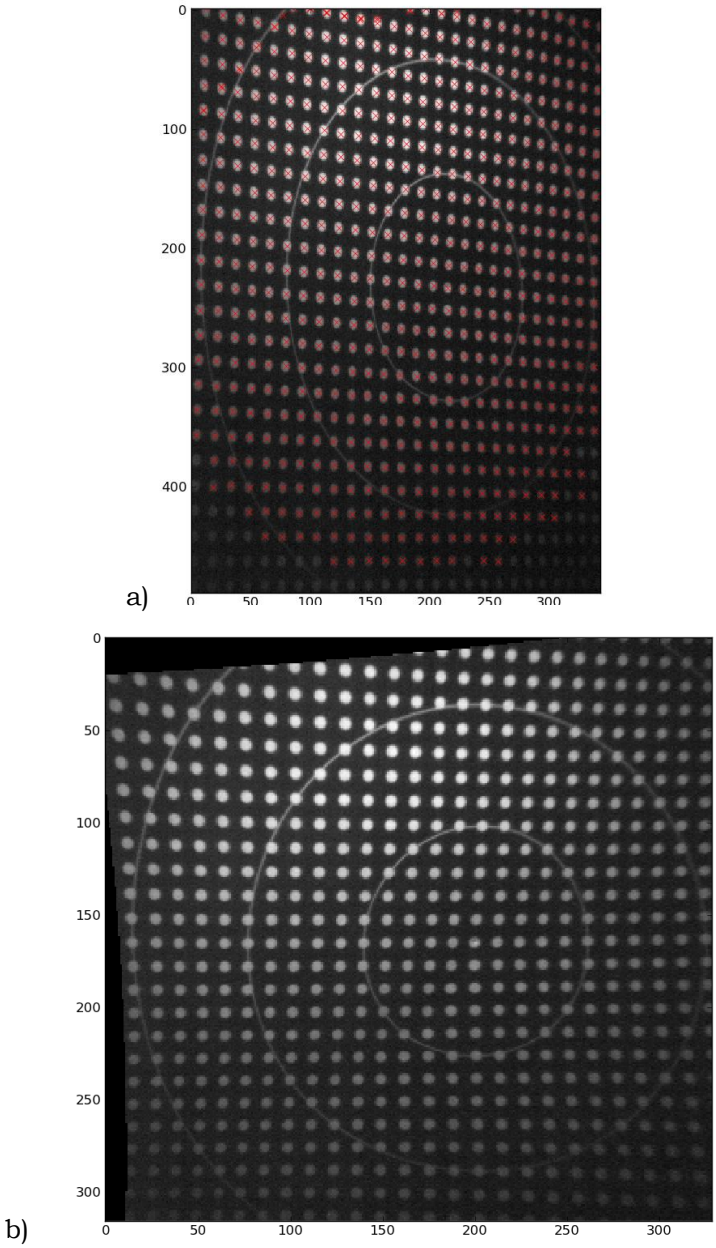
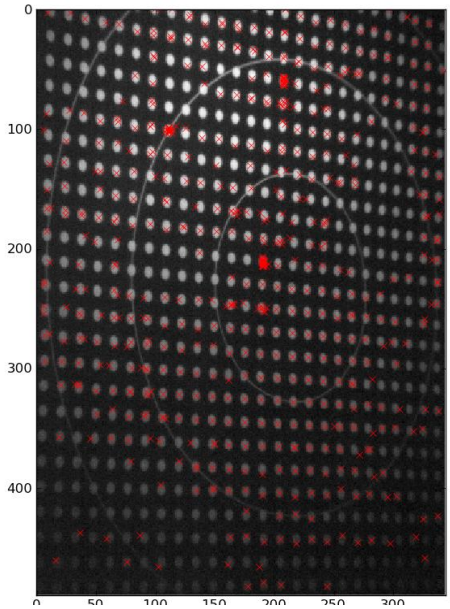


Figure 4-5: a) An image calibration target on the surface of an object. The white spots are the calibration target control points (in this case, points arrayed in a regular square grid, 5 mm between horizontal and vertical neighbour centres) and the red crosses centred on each spot are the centres of the spots found algorithmically through convolution of the entire image with a reference spot kernel. b) The dewarped image from (a) using the found control points. Note the dewarped image takes on a different resolution to accommodate the new shape of the image (in this example, it can be seen that the large faint ellipses in (a) become circles in (b)).

Setting the control point search parameters correctly is important to the impy implementation of the algorithm, as Figure 4-6 demonstrates. Ill-defined search

parameters can return multiple false positives in the image, causing an array of found points with overlaps (multiple points found on the same point) which the mapping part of the procedure attempts to separate, causing a catastrophic breakdown of the dewarping.

a)



b)

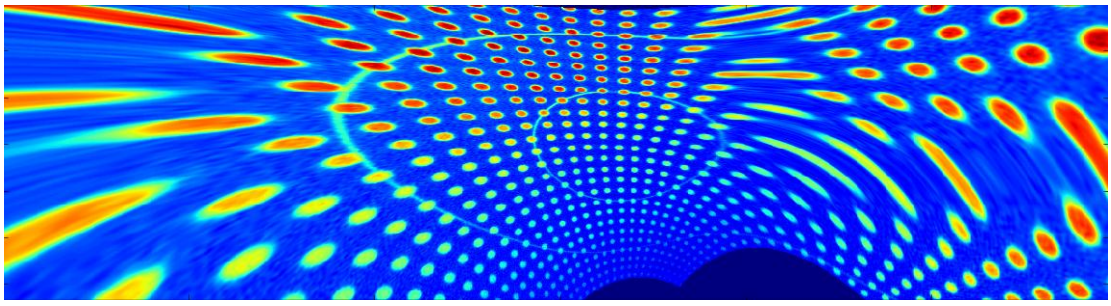


Figure 4-6: a) The same image calibration target image as shown in Figure 4-5 but in this case, the control points have been incorrectly located (note there are clusters of crosses on many of the spots in the centre of the image). b) The resulting dewarped image (in an arbitrary colour map) from (a) displaying catastrophic distortion added as the algorithm attempted to separate the cross clusters.

4.6 Experimental implementation

4.6.1 Introduction to the experimental technique

The experiments, which are described in sections 4.7 and 4.8, were implemented using single component FDM I-PDV. The exact layouts differed for each experiment,

and so will be explained further in their respective sections. A general description will be given here, with the greater part of this section focussing on the interferometer used in the imaging head and the light source used to enable FDM.

The experimental systems all consisted of a light source, an FDM channel generator, a light sheet, a fibre delivered reference beam, and the imaging head with MZI. Image demultiplexing was carried out in post-processing on images captured by the camera in the imaging head.

The two channels in the system were the signal and reference channels, which where respectively the light scattered from the moving flow, and the light delivered into the interferometer directly without a Doppler shift. The reference channel was intended to be simultaneously captured with the signal channel image so to allow the phase change of the scattered light to be measured relative to the un-shifted light, reducing the number of steps required to make a velocity measurement.

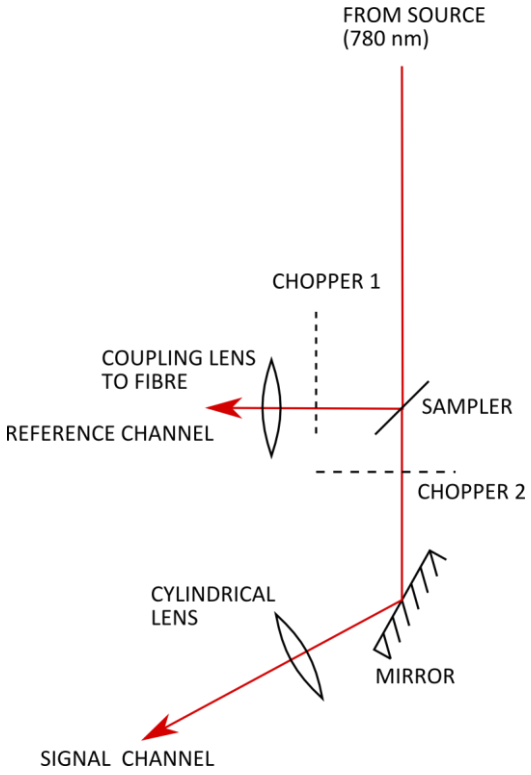


Figure 4-7: Channel generation.

The channels were generated, as shown in Figure 4-7, by splitting the light from a 780 nm Littrow configuration laser diode source (Sacher Lasertechnik Tiger), though a source operating at a wavelength of 532 nm was also used at one stage, into two paths with a beam sampler which split 1% (tuneable to 10% by rotating the incident polarisation state with a half-wave plate) of the light off the main beam. This created

two channels of light, both of which had a beam chopper which rotated at a fixed frequency, uniquely selected for each channel. The low power channel was directed into an optical fibre (Fibrecore SM750) to be used as a reference channel. The remainder of the power continued on past the chopper to the beam forming optics to become the light-sheet. High power was required in the light sheet as the scattering from the particles in the direction of the interferometer was of low intensity, as Mie scattering is predominantly forwards scattered. The observation vector was placed behind the coincidence of the light-sheet to the scattering particles.

The imaging optics, the development and configuration of which is described in the next section, collected Doppler-shifted light scattered from the light-sheet through the imaging lens in front of the interferometer and imaged this onto the camera sensor for post-processing. At the same time, light output from the distal end of the optical fibre was directed onto a diffusing disc that allowed the light to fill a wide area and scatter into many possible angles. The disc was rotating slowly (sub-hertz) to blur out speckle. The Fresnel reflection from a glass slide of this light was imaged by the imaging lens to allow an image of the flow to be viewed simultaneously with the scatter from the reference channel. The ability to view each channel simultaneously on the camera, combined with the modulations applied to each beam via the beam choppers, allowed multiplexing of the images by FDM.

4.6.2 Imaging head configuration

The systems used in the work presented in this chapter varied in the details, but at a basic level consisted of a laser beam split into a reference and a flow signal channel, a moving test object which was imaged onto a camera through a Mach-Zehnder interferometer with a path difference between the two arms.

The exact configuration of the interferometer went through a number of iterations of design before a viable version was found. That is the version described in this section, with another configuration shown in section 4.7 to highlight issues that were raised in the design at that time.

Figure 4-8 shows the interferometer imaging head. It consists of a Mach-Zehnder imaging interferometer in infinity space between a Baumer HXC13-C CMOS camera and an Edmund Optics 25mm f1.4 c-mount imaging lens. The camera sensor captured only light that was allowed through a 780 nm spectral filter (Semrock Brightline FF01-775/46-25) attached to the thread on the camera faceplate. This

acted to filter stray light from entering the camera to prevent it from reducing the useful dynamic range in the images.

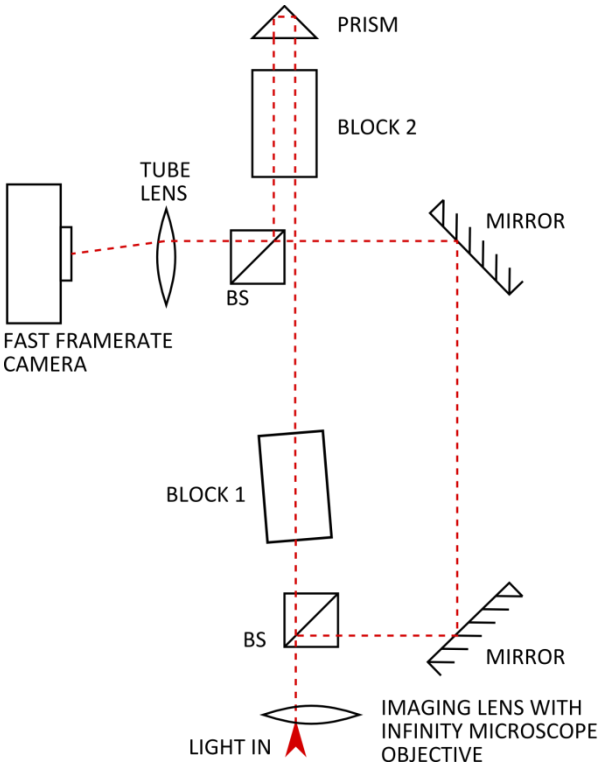


Figure 4-8: Detail of the Mach-Zehnder interferometer used for the practical work in this chapter. The arms of the interferometer are folded to allow for three passes of two glass blocks in one arm while the other arm is completely in air. Block 1 is in single pass configuration and tilting it allows for control of fringes in the interferometer (BS: Non-polarising beamsplitter)

The infinity space was defined between a 180 mm tube lens and an Olympus PlanApo 1.25x infinity-corrected microscope objective. The objective relayed the image of the flow produced by the imaging lens through the interferometer as columns of parallel rays. For each point in the image there was a corresponding column in infinity space, its angle from the optical axis increasing with the radius of the image point from the centre of the image. In effect, each point in the image was described by an angle in the infinity space, along which all rays from that point would travel. This allowed long path lengths in the interferometer with good light collection from the flow. This is illustrated in Figure 4-9.

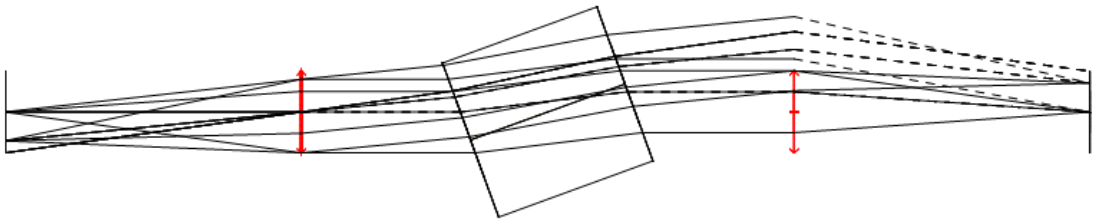


Figure 4-9: A glass block deflecting rays in the infinity space defined between two lenses. The rays originating from the edges are increasingly vignetted due to their angle taking them out of the aperture of the tube lens.

An issue encountered when using a long path length interferometer in infinity space was that, as the path through the interferometer increased, so the centre of each parallel light column increased in distance from the optical axis. This can become an issue when the diameter of the light column meets the edges of components in the system, as, when the rays are clipped from the column, the point to which that column corresponds will become darker in the image when the rays are brought together on the CMOS sensor by the tube lens. This means that, as the paths get longer, the image produced on the camera becomes increasingly vignetted, which reduces the measurable area in the image.

However, the long path lengths available allowed for space inside the interferometer to include large optical components in the light paths without having to be concerned about the issues associated with differential image magnification between arms of the interferometer. In one arm a pair of glass blocks was included to introduce an optical path length difference between the arms of the interferometer, as shown in Figure 4-8. Block 1 (80 mm long) was used to add path length and control the tilt of the fringes, while block 2 (150 mm long) in the same arm, which was passed twice by the light, was used to add further path difference.

There were two conceptual paths for the light in the interferometer, one was the optical path, and the other was the geometric path. The optical path has been described as being controlled by the addition of glass blocks, or regions of higher refractive index, which increased the path travelled. The geometric path length that the light took was controlled by the mirrors in the path without glass blocks (termed the short path here). The mirrors were paired, both in how they controlled the image (see section 4.6.3), and in how they were mounted to allow them to slide together towards or away from the interferometer to change the length of the path the light would travel through air. This allowed the paths in the interferometer to be corrected to match their effective geometric paths (the distance in the two arms the light would travel when the effects of changing the direction of the light travelled by changing

refractive index are taken into account). The geometric path the light travelled in the long path was 665 mm, and in the short path was 520 mm, though as some of the long path was in 430 mm of glass (refractive index 1.5) the optical path length was 884 mm in that arm, leading to an optical path difference of 360 mm between the two arms of the interferometer. This corresponded to a free spectral range of 833 MHz.

The system was designed initially to take measurements with an illumination source operating at a wavelength of 532 nm, but was later converted to use light at 780 nm to take advantage of the available higher power light source (see section 4.7).

4.6.3 Alignment of the interferometer

Alignment of the interferometer is achieved through a number of processes performed in sequence, and also some which can be performed through the duration of use to facilitate minor corrective alignments.

Firstly, for the initial alignment of the interferometer, the components are placed on the optical bench in position by hand in their approximate positions and a visible light source, such as a He-Ne laser, is used to orientate the optical surfaces. The unexpanded beam is coupled into the interferometer along the optical axis, this axis being defined by placing pinhole apertures in the arms and the laser beam is steered through their centres. Then, by monitoring the back-reflections of this light from the various interfaces and overlaying these reflected beams on one another at a back plane near the laser aperture, the angles of components can be made close to parallel. This step works well for the alignment of transmissive components with optical surface vectors that should be aligned to the optical axis, such as the lenses, beam splitters, turning prism and glass blocks, but does not provide a mechanism for controlling the overlay of the image circles (vignetting) from the two arms, nor for controlling the positions and rotation of the images within those circles.

The overlaying of the image circles, and the rotation and alignment of images within those, are mostly controlled by two pairs of components within the interferometer. The first pair consists of the two beamsplitters, and the other pair consists of the two turning mirrors. At the initial beamsplitter, the light is split into the two arms, one of which is reached through transmission of half the light through the cube, while the other is reached when half of the light in the cube is reflected at the internal interface and travels perpendicular to the transmitted light in the other arm. The light in the two arms propagates along their respective arms of the interferometer before being recombined at the second beamsplitter. In this case, the light from the arm that was

reflected at the first beamsplitter is now transmitted through the second beamsplitter to the tube lens and onto the camera, while the light which was transmitted at the first beamsplitter is now reflected at the second beamsplitter to the tube lens and then the camera. This pairing works well when the entry faces of the beamsplitters are well aligned to the optical axes, but, as in the arrangement shown in this chapter, it was not possible to use a cage mounted second beamsplitter as it had to be placed close to the return path in double pass glass block and thus the mount would impinge on the beam-path.

With the exception of the glass blocks and prism, the optics were mounted in a cage system. This type of mounting arrangement places, in theory, the optics on a common optical axis defined by the rails on which faceplates slide along with optical components attached in their centres. Unfortunately, with the long path lengths in the system, the optical cage mounting system employed (a combination of Thorlabs 30 mm optical cage system and Linos Microbench) did not remain in perfect alignment due to the cumulative effects of small tolerances and the presence of sections where large components had to be mounted externally to the cages, breaking the rigidity of the rod supports. Thus, it was necessary to mount the beamsplitters with appropriate adjustment capability to compensate for image misalignments. Aligning the light in the two paths required two types of adjustment to be made to the beamsplitters; rotation about the input face to control the vertical angle of reflected light, and rotation in the plane of the optical bench to control the horizontal angle of the reflected light. The transmitted light was not as strongly influenced by the alignment changes as was the reflected light, so adjustment of the beamsplitters focused on the control of the alignment of light on reflection. As the light in the short path was sent there in reflection from the first beamsplitter, it was used to control the light at the start of the short path, which was in transmission at the second beamsplitter. The alignment of light transmitted at the first beamsplitter into the long path was corrected at the end of this path by the second beamsplitter which reflected it to the camera.

It is worth noting that this beamsplitter alignment technique works well to align the interferometer for a single input port to the MZI, but it was discovered that if both of the input ports were used to make measurements, for example putting the reference channel through the other port so to remove the need for the glass slide, then the points imaged through each had to be in conjugate input planes, otherwise the images from each port would have slightly different paths through the interferometer and so evaluate to different phases. This effect was seen using the tilt fringes with this method, these showing slight differences in the fringe frequency when the fringes were

evaluated with a 2D FFT, as the fringe peaks in Fourier space would be shifted in the window, introducing low frequency spatial distortions to velocity measurements.

The image circles can be aligned by using the laser beam, expanded and collimated, blocking one arm and then swapping to the other and checking for clipping occurring from any misaligned components that appear as shadows in the expanded beam. While that would allow large misalignments of individual components to be corrected, image circle alignment is perhaps more easily and accurately achieved by actually imaging through the interferometer onto the camera and making adjustments based on viewing the images themselves.

Figure 4-10 shows images obtained from the interferometer under a number of misalignment states. Part 1 of that figure shows a subjective speckle pattern from a diffuser's surface imaged through the interferometer when the two arms are well aligned. Part 2 shows what happens to the image when the speckle is blurred by rotating the diffuser.

Of great utility is the use of a real-time 2D FFT of the image on the camera. When this is used in conjunction with a live image of a speckle pattern, precise alignment of the images from both arms of the interferometer is made possible to sub-pixel resolution. It is important to use an image of laser speckle, or a similar type of pattern with a high density of speckles, for this, as it will have the effect of producing a similar pattern in the Fourier domain (for an example, see Figure 4-10 (1) and Figure 4-11 (1)). The FFT image displayed is the 2D power spectrum of frequencies in the real-space image, arranged so that at the centre is the zero-frequency DC term for the background intensity, and in all angular directions from this point can be considered the equivalent one-dimensional FFT of all the pixels lying along a line at that angle. If the features of the speckle pattern are both well defined and small, then these features are described by a spread of spatial frequencies up to the Nyquist limit. When the 2D power spectrum is displayed, the distribution of frequencies has the effect of filling the frame of the 2D FFT image with a similar speckle image.

The 2D FFT was used extensively in image alignment, as, when the images from the two arms of the interferometer were overlaid well, a clear 2D FFT of the images would be displayed (for an example, see Figure 4-10 (1), and Figure 4-11 (1) where multiple spots can be seen due to reduced dynamic range used to bring out the background speckle features). A small misalignment between the two images would reduce the quality of the 2D FFT in a very deterministic way based on the type of misalignment present. Therefore, the initial approximate alignment of the images would be performed by eye, by looking at the images from the two arms and overlaying the

features. Once this was done, the speckle in the two images would begin to interfere and this would be apparent in the FFT. In this, lateral misalignments between the two images would appear as fringes across the frame, perpendicularly aligned to the image misalignment. As the lateral displacement between the images increased, the frequency of the fringes increased until the point where the speckle de-correlated and the fringes would vanish (an example of this can be seen in Figure 4-10 (3-5) and Figure 4-11 (3-5)).

Another strong effect was that of relative image rotation, which would allow a region in the images to be well aligned, but would show increasing misalignment as the distance increased radially. Rotational misalignment manifested as a loss of high frequency information in the FFT, where with increased rotational misalignment came correspondingly reduced higher frequency information. If lateral misalignment was present in addition to the rotational misalignment there would be fringes present in the FFT as well, though these would only be visible in the region where the frequency information was present from the rotational interference (for an example, see Figure 4-10 (6-7) and Figure 4-11 (6-7)).

Realignment of the images in the interferometer involved the adjustment of the screws on the back of the holders of the paired mirrors. Lateral misalignment would be corrected by adjusting the screws on both mirrors: first by picking a mirror and adjusting the screw that moved the image along the plane of the bench, and then again using the corresponding screw on the other mirror to return the image back along the plane of the bench. Vertical misalignment could be corrected by adjusting one of the 45° angle screws on either mirror to overlay the images. Note that the use of the 45° adjustments introduces a rotational misalignment. Rotational misalignments were corrected by using the 45° adjustment screws in pairs. The pairings were the top and bottom screws of one mirror then the next for a given rotational direction, and vice versa for the opposite rotation. It was indeterminate which pair should be used initially, but improvement in the misalignment was used as the indicator of the correct choice. Using the first screw of the pair, the image would be moved at a 45° angle across the frame and then returned back to the centre using the other screw of the pair.

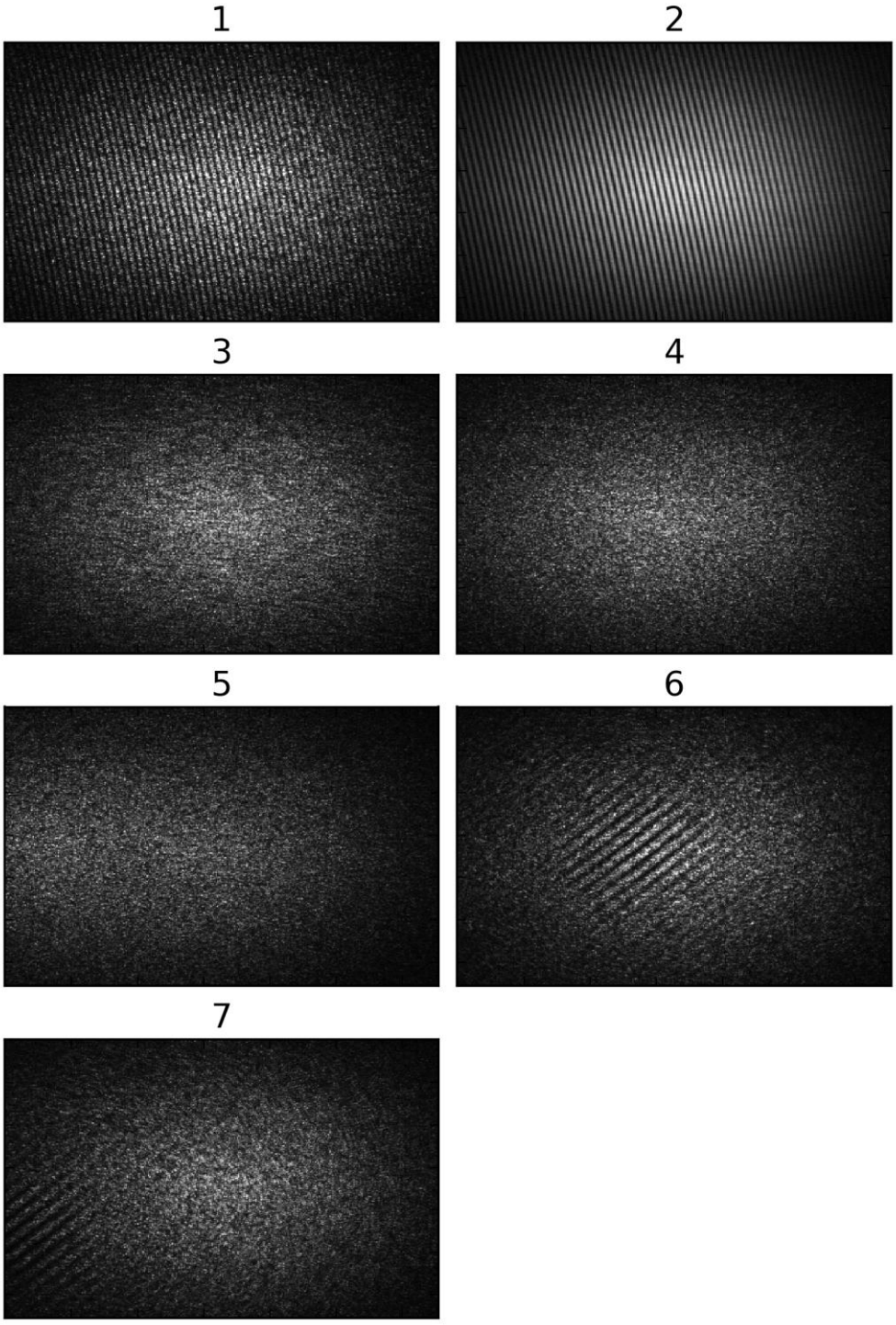


Figure 4-10: Images of fringes obtained from the MZI. (1) Perfectly aligned speckle patterns, (2) well aligned speckle patterns with rotating diffuser, (3) slight lateral misalignment, (4) large lateral misalignment, (5) completely de-correlated speckle pattern with half-frame width lateral misalignment, (6) rotationally misaligned speckle patterns, and (7) rotationally misaligned speckle patterns with lateral shift.

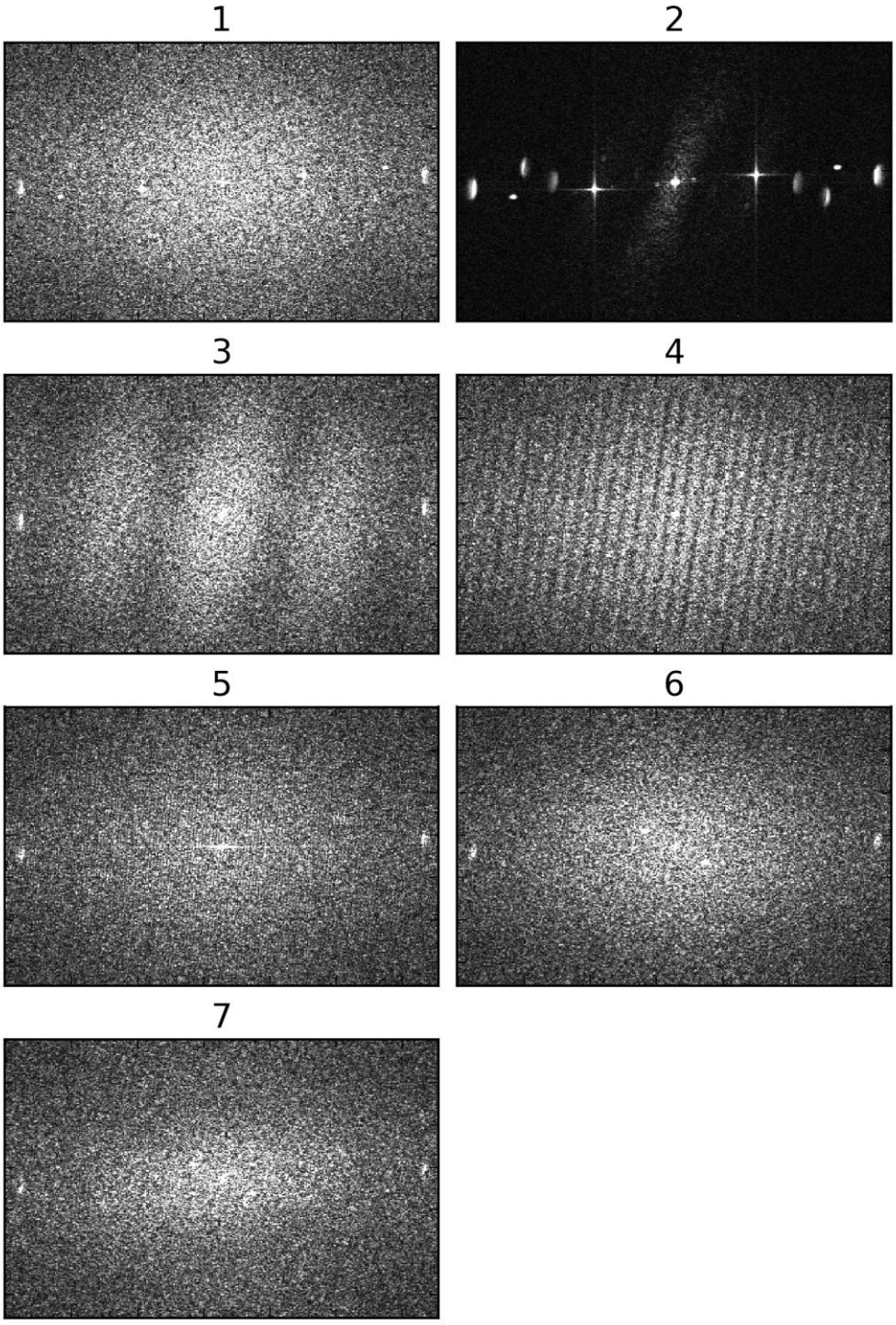


Figure 4-11: Spatial frequency space images of the fringes shown in Figure 4-10. (1) Well aligned speckle patterns, (2) the same as part 1 with a rotating diffuser, (3) slight lateral misalignment, (4) large lateral misalignment, (5) completely de-correlated speckle pattern with half-frame width lateral misalignment, (6) rotationally misaligned speckle patterns, and (7) rotationally misaligned speckle patterns with lateral shift.

4.7 Measurement of the velocity of a disc

4.7.1 Introduction

The first experiment performed with FDM I-PDV was the measurement of the velocity of the side face of a disc rotating about a fixed axis.

The optical layout was as described in section 4.6.1, with the moving flow being simulated by a rotating disc with a highly scattering surface coating of dust particles. This coating was created by lightly scoring the white painted surface of the disc with an abrasive pad.

A disc was chosen as the first object to measure because the velocity of a disc is well defined and therefore would quickly confirm that the system was working. The speed at any point on the surface of the disc is given by:

$$v(r) = 2\pi r f_r \qquad 4\text{-}13$$

where $v(r)$ is the speed at a given radius, r, on the surface, and f_r is the frequency of rotation of the disc (RPM/60), and the corresponding direction component is perpendicular to the radius in the direction of rotation.

4.7.2 Procedure

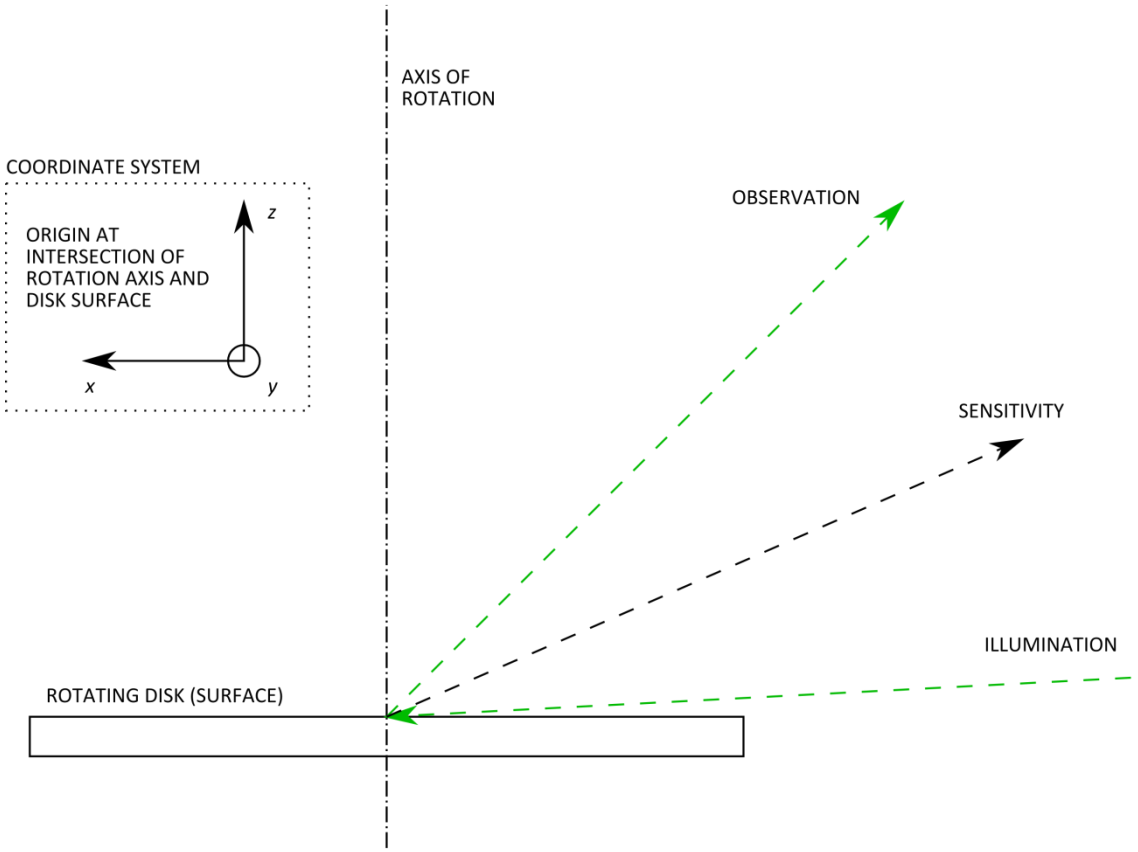


Figure 4-12: The disc used in this experiment, defining the coordinate system and showing the incidences of the vectors on the surface

The disc was orientated with respect to the observation and illumination vectors as shown in Figure 4-12, which defined the coordinate system for the experiment (this coordinate system was an output from the image dewarping that was applied to transform the coordinates from positive z on the observation axis to be collinear with the rotation axis of the disc). Light was incident at a grazing angle to the surface of the disc, which had its axis of rotation at 45 degrees to the observation axis in the plane defined by the observation and illumination vectors. This meant that the illumination light-sheet was incident at nearly a right-angle to the disc surface, bringing the disc surface as close to the sensitivity vector in the system as was possible. Any closer to the sensitivity vector would necessitate illumination through the back of the disc, impossible due to the opaque material from which the disc was constructed.

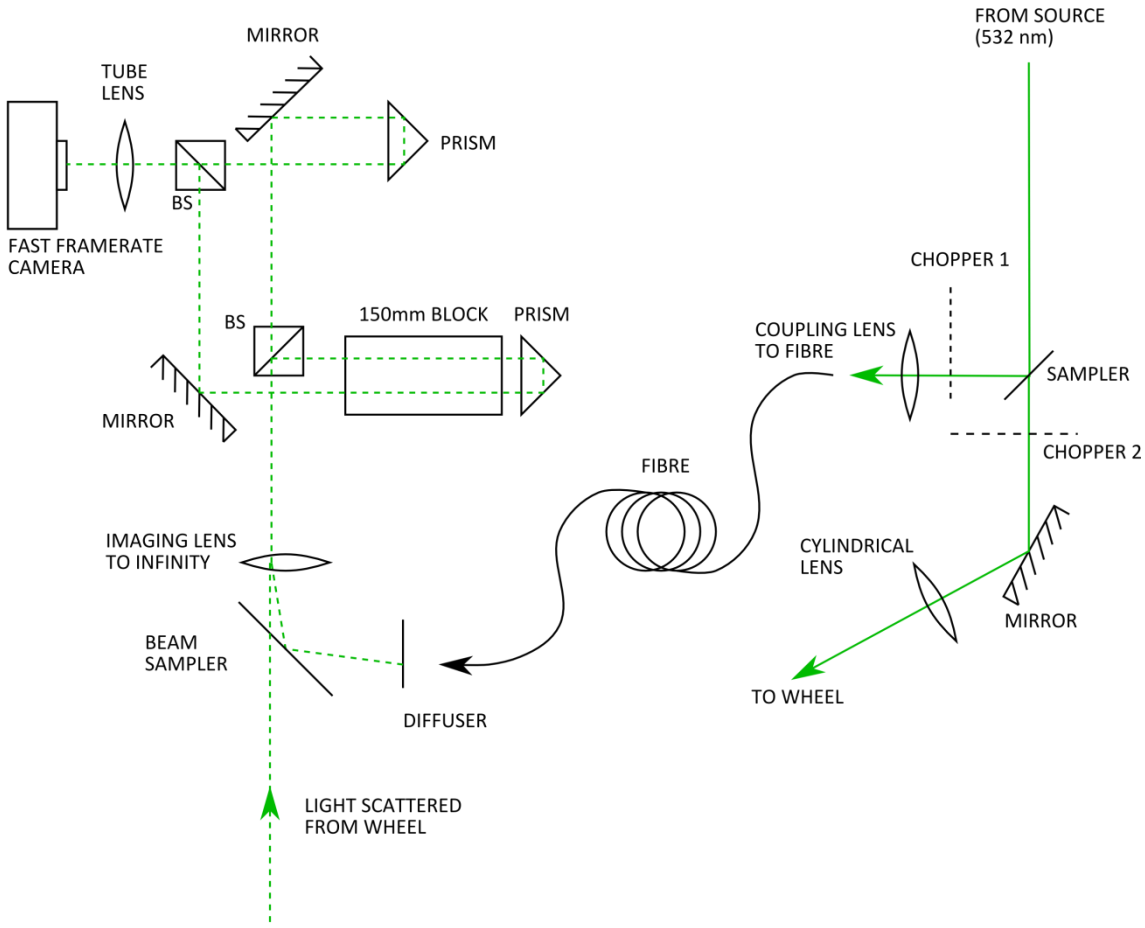


Figure 4-13: Initial configuration of the I-PDV imaging head and illumination optics. The two prisms in both arms of the Mach-Zehnder interferometer were used to turn the light, though this arrangement was replaced with the revised configuration of the interferometer (shown in Figure 4-14) using only a single turning prism in one arm and a paired set of turning mirrors in the other arm to allow for greater image control. (BS: Non-polarising beamsplitter)

The interferometer was designed initially around a MZI configuration where the beams were turned in both paths by prisms, and in one arm a glass block was included to increase the optical path length. This configuration is shown in Figure 4-13. The system was designed to use the Fourier transform method for phase extraction, and tilt fringes were designed to be introduced by tilting one of the prisms off the optical axis. This introduced issues however of steering the images within the image circles so that the images from both arms were sheared from one another. Another method that could be used to introduce the tilt fringes involved the movement the image circles

using the mirrors, but this translated the image circles from concentricity, such that fringes were visible over only a limited area of the image. These two techniques could be combined to allow both the fringes to be introduced and to steer the images back to being overlaid on one another. However, this technique was not a perfect compromise as the images and the image circles would always be shifted relative to each other as it was not possible to align mechanically the optical mounts to allow them to translate optics in the same planes as each other. Thus an image rotation was also introduced into the system. As the system was designed as shown in Figure 4-13, this rotation could not be removed with ease.

To facilitate easier control of the image when setting-up the interferometer, it was redesigned to the form shown Figure 4-14. Two changes were made; firstly the interferometer was reconfigured, and then the illumination source was changed.

As per the description in section 4.6.2, the interferometer was redesigned with the two mirrors paired in the short path to replace the turning prism and allow fuller control of the images. The prism and glass block remained in the long path. The redesigned form provided space in the long path to include an additional glass block, which was used to add the tilt fringes. Any image offsets were controllable with the paired mirrors, correcting for the offsets by using the DFT of the image as described in the alignment section (section 4.6.3), allowing precise correction for the image.

At the time of the redesign, the illumination source was a green DPSS laser operating at 532 nm (Coherent DPSS 532-300), selected due to the convenience of working in the visible regime, and the ready availability of optical components. It was found that the power of this laser on scattering was insufficient to produce adequate filling of the digital levels of the CCD camera on the edges of the image circles after traversing the long infinity space inside the interferometer. This was due to the divergence of the point ray columns causing them to be lost by the end of the interferometer, when they were already of low intensity on being imaged by the imaging lens.

To combat the low light levels, the light source was replaced by the 780 nm Littrow configuration laser diode source (Sacher Lasertechnik Tiger) which had an output power of up to 1.2 W in comparison with the 200 mW CW output from the Coherent DPSS 532 nm source. This was expected to produce greater filling of the digital levels in the camera to provide greater fringe resolution. However, it was found that the interferometer produced low visibility fringes at this wavelength, which stopped the phase calculations functioning accurately. The reason for this was found to be that the light was being absorbed by the aluminium mirrors in the short path, giving an 85% reflectivity and the beamsplitters' split ratios were no longer 50:50, but rather

70:30. To compensate for this, the components were replaced by wavelength-appropriate optics. The aluminium mirrors were replaced with silver mirrors which provided >98% reflectivity. The beamsplitters were replaced with models coated for the correct wavelength. This was a significant improvement, as it took the light in the two arms to near equal intensities on recombination at the second beamsplitter, improving the visibility of the fringes.

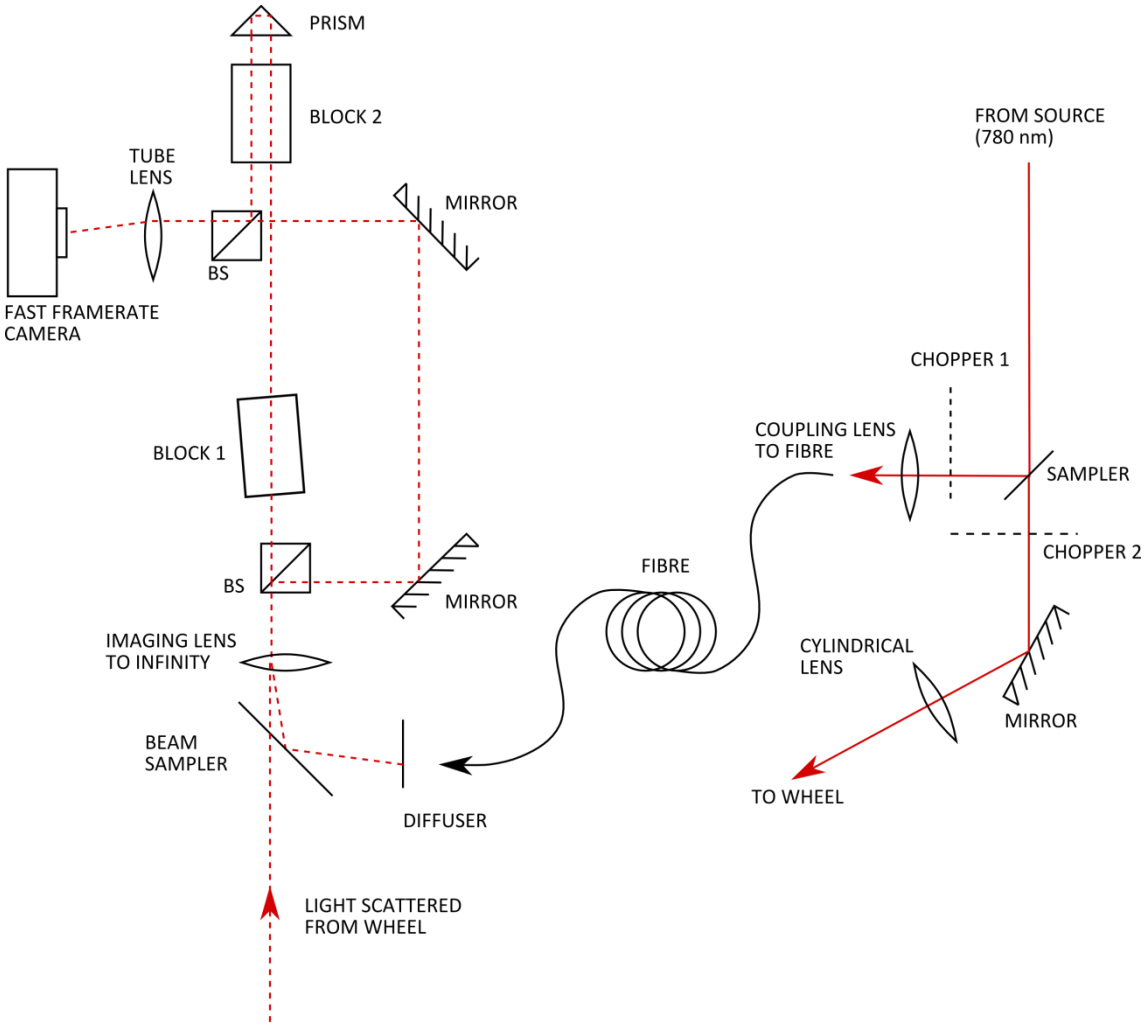


Figure 4-14: Schematic of the revised experimental arrangement, showing the Mach-Zehnder interferometer and the illumination components. The paired mirrors allowed for image steering, and block 1 allowed the introduction of tilt fringes into the system. (BS: Non-polarising beamsplitter)

Images of the disc were taken, in time-series of 256 frames, while illuminated when at rest and again while moving, allowing three measurements to be made of velocity. One

method used TDM to make two measurements, one using an image of the flow at rest (though for practical purposes the resting flow was actually at a low velocity in order to blur out speckle) for a reference image, and another measurement was made using the fibre reference. The remaining measurement used FDM to multiplex the fibre reference channel and the measurement channel. The FDM measurements were demultiplexed to obtain the reference and signal images, while the TDM measurements were completed in two banks each, with a times series for the signal and the reference images, which were obtained by averaging the image time-series. A 450 x 50 pixel Hamming window, centred on the positive frequency peak, was applied to the Fourier space images to allow the determination of the phase from the carrier fringes (there were approximately 55 fringes horizontally across the field of view). This window had an aspect ratio designed to select for the directionality of the peak in the Fourier spectrum produced by the fringes, while the smooth profile of it allowed the image to gently reach zero at the edges and reduce peak broadening associated from abrupt truncation of the data.

4.7.3 Results and discussion

Figure 4-15 shows the velocity determined from the FDM measurement of the spinning disc, and Figure 4-16 shows the theoretical velocity expected from the same configuration.

The theoretical values were derived from measurements of the rotation of the disc using a light gate and a piece of reflected tape affixed to the rear of the disc, allowing RPM measurements to be obtained. The RPM value was mapped to a velocity vector for a given radial distance from a centre of rotation and the array of these values was distorted to take account of the effects of perspective in the system, then were multiplied by the measured sensitivity vector to yield the theoretical values.

The FDM measurement and the theoretical velocity fields show good agreement, though with two small systematic errors on the FDM result. The first error is a small offset of approximately 6 ms^{-1}, potentially a result of the fringe drifts in the interferometer, from thermal expansion in the components and from temperature drifts (and associated wavelength changes these induce) in the laser diode, and the effect of the light travelling slightly different paths through the interferometer and experiencing slightly different path differences, producing a slight phase offset between the channels. The second error appears as a slight under-evaluation of the velocity across the field of view, which appeared to follow the intensity distributions of

the channels. A histogram is shown of the difference between the model and the theoretical results in Figure 4-17. It shows clearly the offset on the measurements.

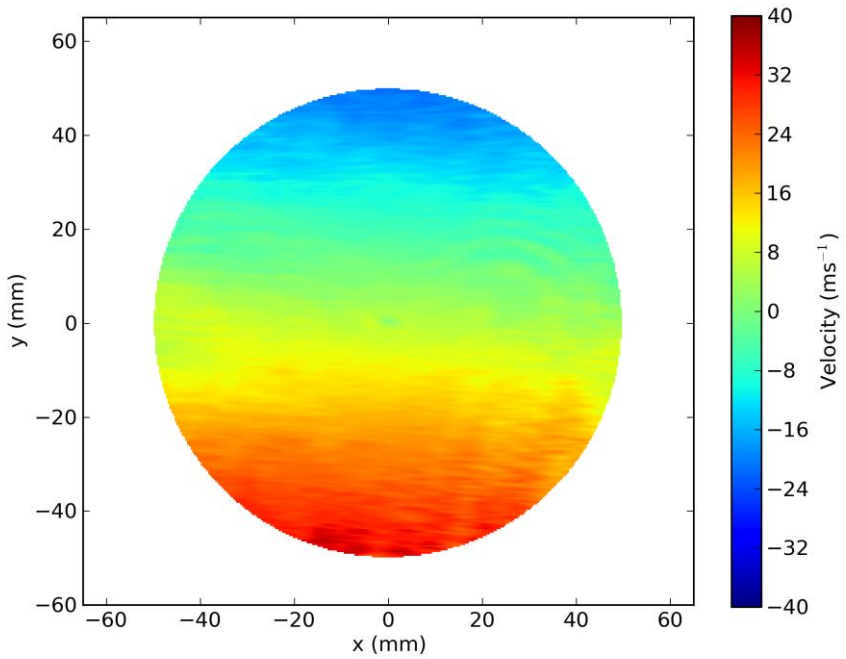


Figure 4-15: The velocity on the surface on the disc measured using FDM.

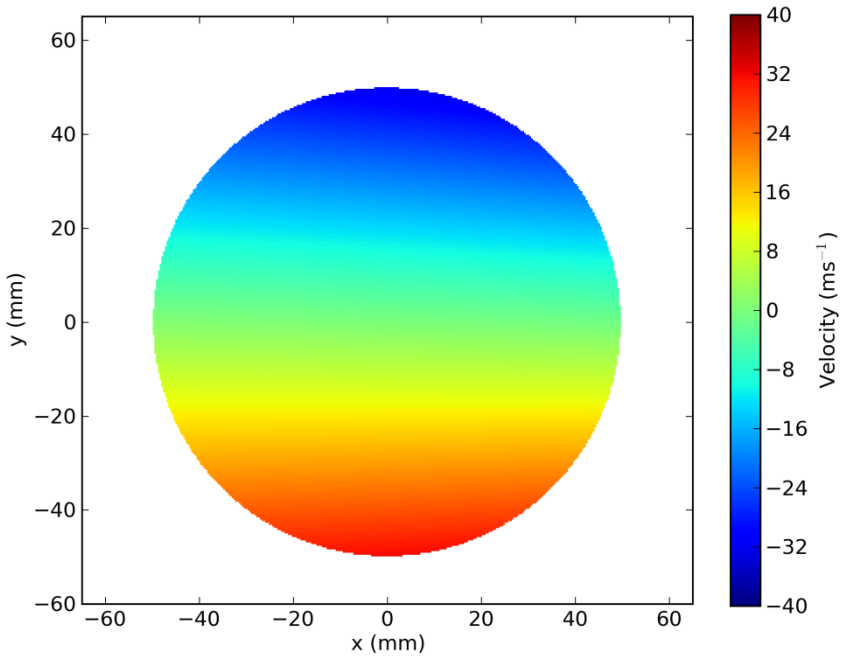


Figure 4-16: Theoretical velocity distribution on the disc surface.

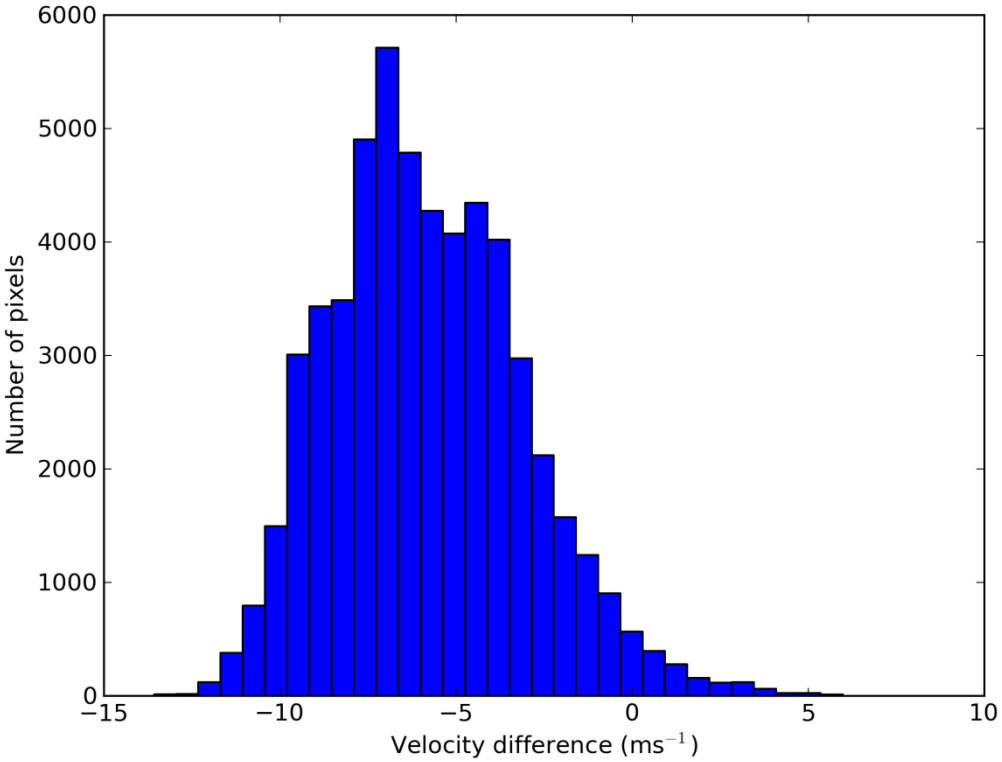


Figure 4-17: Histogram plot of the difference in velocity measured using FDM and the modelled result.

The three measurements made of the velocity, using TDM with the disc and fibre references, and with FDM using the fibre reference, are plotted together in Figure 4-18. A number of conclusions can be drawn from the figure. Firstly, it is clear that the two TDM measurements have a large offset error, and this is due to the fact that the TDM technique has a time delay between capturing the signal and reference images, during which any fringe drift has a chance to accumulate. Any measurement-induced fringe shifts can be small in comparison to the drift, and so the drift effect dominates the measurements in TDM and compensation efforts must be made to remove it in post processing. The FDM measurement has a much smaller offset error, so while not perfect, the FDM result is more representative of the theoretical velocity.

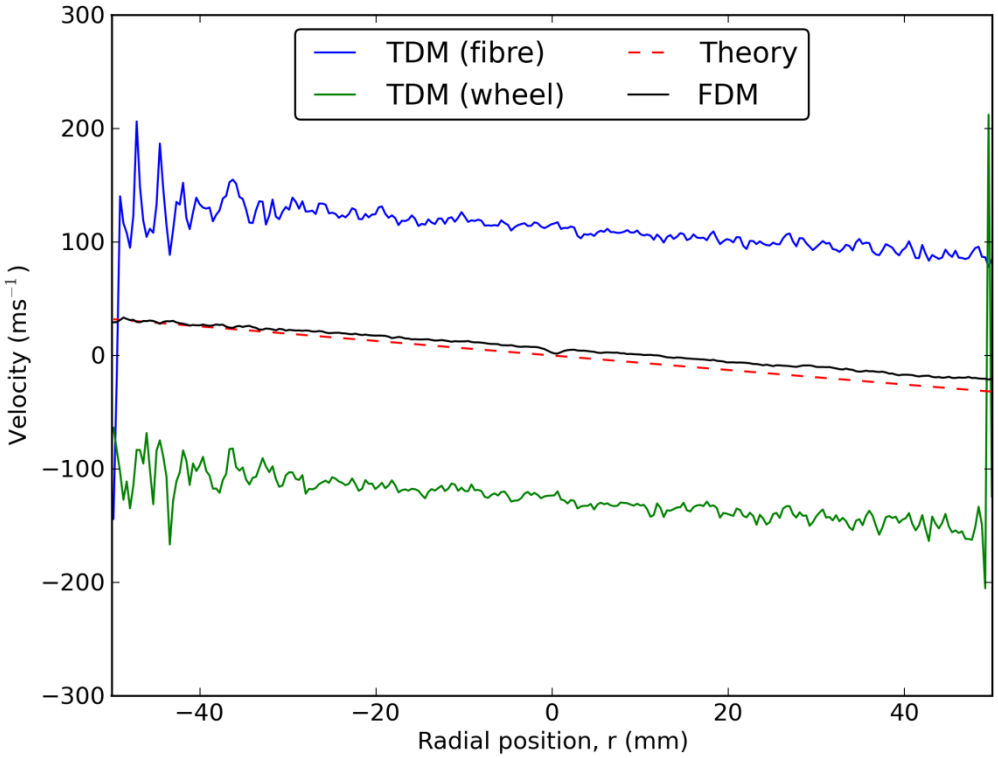


Figure 4-18: Plot of velocity profiles taken from the vertical diameter of the disc surface using three techniques: FDM, TDM with the disc revolving slowly as the reference, and TDM using the same fibre reference as the FDM measurement. A large offset can be seen in the two TDM measurements in comparison to the FDM result, due to the interferometric phase drifting constantly in the time difference between measuring the reference and the signal channels.

It is clear too that the FDM measurement has a much higher signal to noise ratio, appearing smoother by eye than the TDM measurements. The FDM result therefore shows detail of the moving object, such as the pinhole at the centre of the disc where the light levels were low, which is manifested as a noise spike in the velocity.

The under-evaluation of the FDM velocity is shown as lower gradient line in the figure. It is possible that the noise in the TDM results masks the same effect in the two TDM measurements, as the gradient difference can be contained entirely within the noise. Noise reduction has been performed on both the TDM and FDM measurements, in time averaging the TDM measurements and the demultiplexing of the FDM results, but the FDM measurements are shown to be more effectively liberated from noise.

In the low light regions, noise can be seen to increase in all measurement sets, though the effect is more pronounced in the TDM measurements, with noise covering a 100 ms^{-1} range.

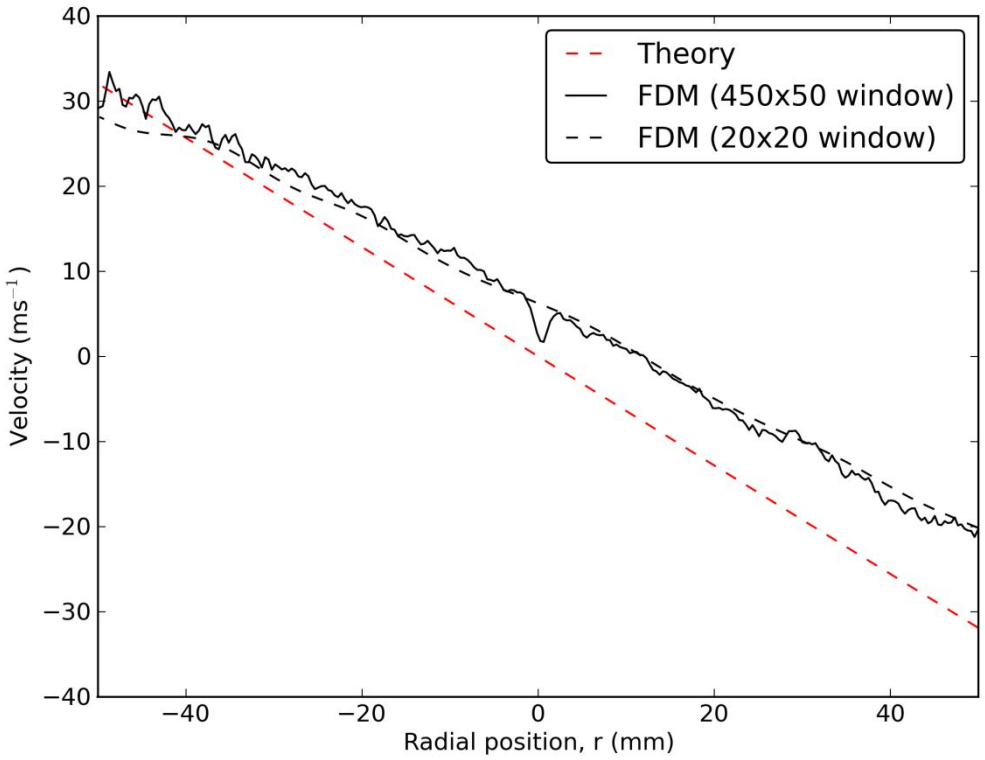


Figure 4-19: Plot of the FDM result with different filtering windows applied during the carrier fringe method processing for the phase. The filter window sizes are given in the legend, and are the dimensions in pixels of the Hamming filter used to isolate the carrier fringe peak.

The filtering used in processing the results can affect the final results. This is shown in Figure 4-19 where two profiles are shown of the FDM measurement, processed twice using different peak isolation window functions in the carrier fringe method. The results shown in the figure include measurements already presented in this chapter using the 450 x 50 pixel Hamming window, and another set calculated using a much smaller Hamming window of 20 x 20 pixels, effectively isolating the peak with a low-pass filter. This has the effect of removing the spatial noise from the velocity image, but it comes at a cost of smoothing out detail and it can be seen in the figure that where the noise is higher, at around $r < -35$ ms^{-1}, the velocity deviates quite suddenly from the trend. This shows that the window function must be chosen with a view to not over-smooth the data and alter the results.

4.8 Measurement of the velocity of an air jet

4.8.1 Aim

In order to undertake measurements of a gaseous flow, the experimental arrangement was redesigned to allow the insertion of an air jet nozzle and particle seeder. The results of this experiment are presented below.

4.8.2 Air jet modelling

The air jet was modelled as a circular free jet in a compound flow. The structure of such a flow was described by Rajaratnam [73] and a set of empirically derived equations were used as the basis for modelling the velocity profiles captured experimentally, and these are described in this section. The application of these equations to the experimental results can be seen in section 4.8.6.

The structure of a circular free jet consists of three distinct regions: the region inside the nozzle, the flow development region just outside the nozzle, and the fully developed region.

The area of the flow to be measured with the PDV technique was the flow development region, chosen as there was a well-defined structure to the velocity of the flow in this region and models were available to describe this structure. This region extends from the end of the nozzle to a point approximately ten nozzle radii downstream, and is the transitional region between the flow inside the nozzle and the fully formed jet. It contains a zone of constant velocity, called the potential core, which is gradually reduced in dimensions from the diameter of the nozzle ($2r_0$) down to zero as the fluid expands, eventually becoming the fully formed jet. The rate of decrease of the constant velocity region radius is given by equation 4-14 for r_1, where x is the position downstream from the nozzle, and m_1 is the gradient of the decrease with respect to the downstream position.

$$r_1 = r_0(0.95 - m_1 x) \qquad 4\text{-}14$$

The boundary edge, r_2, of the expansion of the jet into the surrounding medium of the same fluid is given by equation 4-15, where m_2 is again the gradient of the line with respect to the downstream position.

$$r_2 = r_0(1.04 - m_2 x) \qquad 4\text{-}15$$

The exact point of origin of r_1 and r_2 should ideally be at the end of the nozzle. In practice, this varies with experimental set-up, and is often called the virtual origin of the nozzle. It is usually located inside the nozzle, but can also be located in front of the nozzle.

The gradients m_1 and m_2 are determined from empirically derived fits to results obtained by Rajaratnam. In that work, four points were obtained for both gradients at four values of Λ, which is the symbol given to the ratio of the velocity of the fluid in the flow to that of the background fluid. In the model developed for use in analysis of the PDV results obtained in section 4.8.6, the values of the gradients at Λ values between the measured points were obtained by linearly interpolating between them. The possible values of the gradients are shown in Figure 4-20.

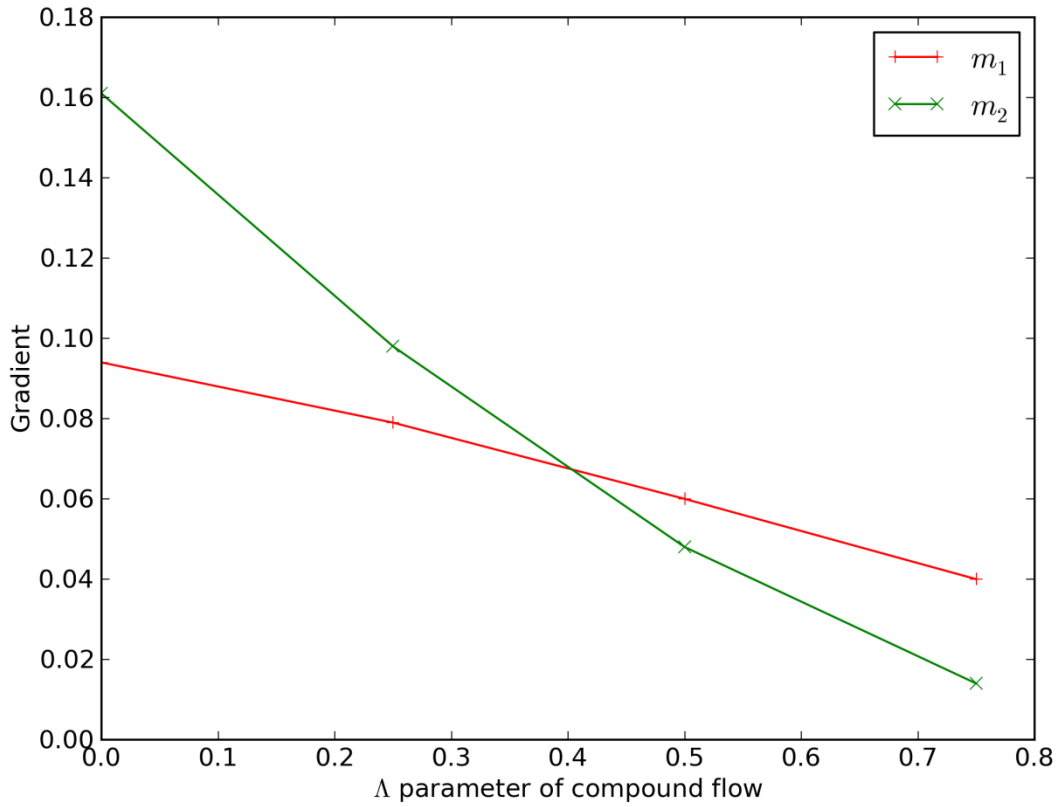


Figure 4-20: Plot of the linear interpolation (lines) between the results of Rajaratnam (markers) used to determine the values of m_1 and m_2.

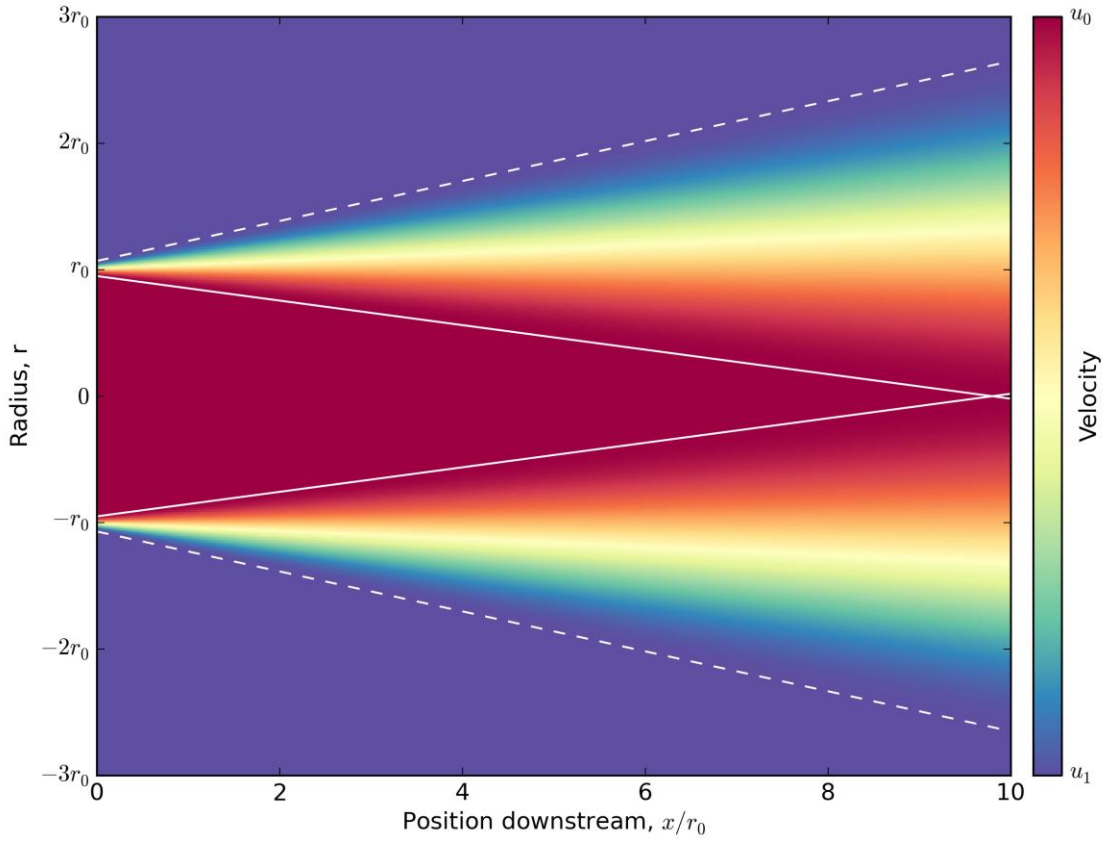


Figure 4-21: Plot of the velocity structure of the flow development region of a circular free jet as a compound flow, plotted as a function of position downstream of the nozzle and radius across the flow. With increasing r, the solid lines mark the boundary between the potential core and the shear region, and the dashed lines indicate the boundary between the shear region and the ambient flow.

The velocity of the jet in the development region is structured into three transverse zones. The zones, shown in Figure 4-21, are the potential core from $r = 0$ to $r = r_1$, the shear layer between $r = r_1$ and $r = r_2$, and then the ambient fluid for $r > r_2$. The core, from centre of the flow to the r_1 line, is the region of constant velocity, which is the initial velocity at the origin, u_0. In the shear layer, the velocity, u, decays as a cosine function with increasing radius, r, such as that shown in equation 4-16 for the case of a circular free jet in a compound flow.

$$u = \left\{\left[\frac{1}{2} - \frac{1}{2}cos\left(\frac{r_2 - |r|}{r_2 - r_1}\pi\right)\right](u_0 - u_1)\right\} + u_1 \qquad 4\text{-}16$$

The velocity u_1 is the velocity of the ambient fluid flow into which the jet is being emitted. The dimensionless velocity structure of the flow development region is shown in Figure 4-21 for a plane through the jet intersecting a diameter of the nozzle and the axis of the jet. The velocity of the flow is represented by the colour map, and ranges

from the velocity of the fluid at the end of the nozzle, u_0, to the velocity of the ambient fluid, u_1. The solid white lines are the r_1 lines, and the dashed white lines are the r_2 lines. From $r = 0$, the constant velocity of the potential core can be seen between the solid white lines. Out from that, between the solid and dashed white lines, is the shear layer where the jet is forming, described by equation 4-16, and beyond the dashed white lines is the ambient flow. The jet is shown travelling from left to right, with $\Lambda = 0$, the nozzle positioned at $x/r_0 = 0$, and the end of the development region coming where the solid white lines cross the jet axis at $r = 0$. Were Λ to be increased, the crossover point of the r_1 lines would be at an increased x/r_0 value.

4.8.3 Experimental layout

The schematic of the experimental configuration is shown in Figure 4-22. The illumination vector from the light sheet was orientated at right angles to the observation vector. This had the effect of positioning the sensitivity vector on the bisecting angle, which lay at 45° between the two vectors.

Illumination was provided by Sacher Lasertechnik's Tiger model Littrow configuration external cavity diode laser, operating at 780 nm. The power at the laser aperture was 1.2 W, which dropped to 810 mW after transmission through the external isolator then half wave plate positioned close to the laser exit aperture in order to protect the source from back reflections. This was then split into the two channels by a glass wedge beam sampler (Thorlabs BSF10-B), which was tuned to reflect 1% of the light into the fibre channel with p-polarised incident light. The rest of the light was transmitted through to the light-sheet optics.

The light-sheet optics consisted of a -50 mm focal length cylindrical lens and a 250 mm spherical lens in a telescope arrangement, placed 760 mm away from the image plane of the camera lens at the input to the MZI. The cylindrical lens expanded the incident light in the vertical axis, while maintaining the beam width in the horizontal axis. The height of the light-sheet was 50 mm at the image plane, collimated in that axis by the spherical lens, allowing measurement of flow structure inside that region.

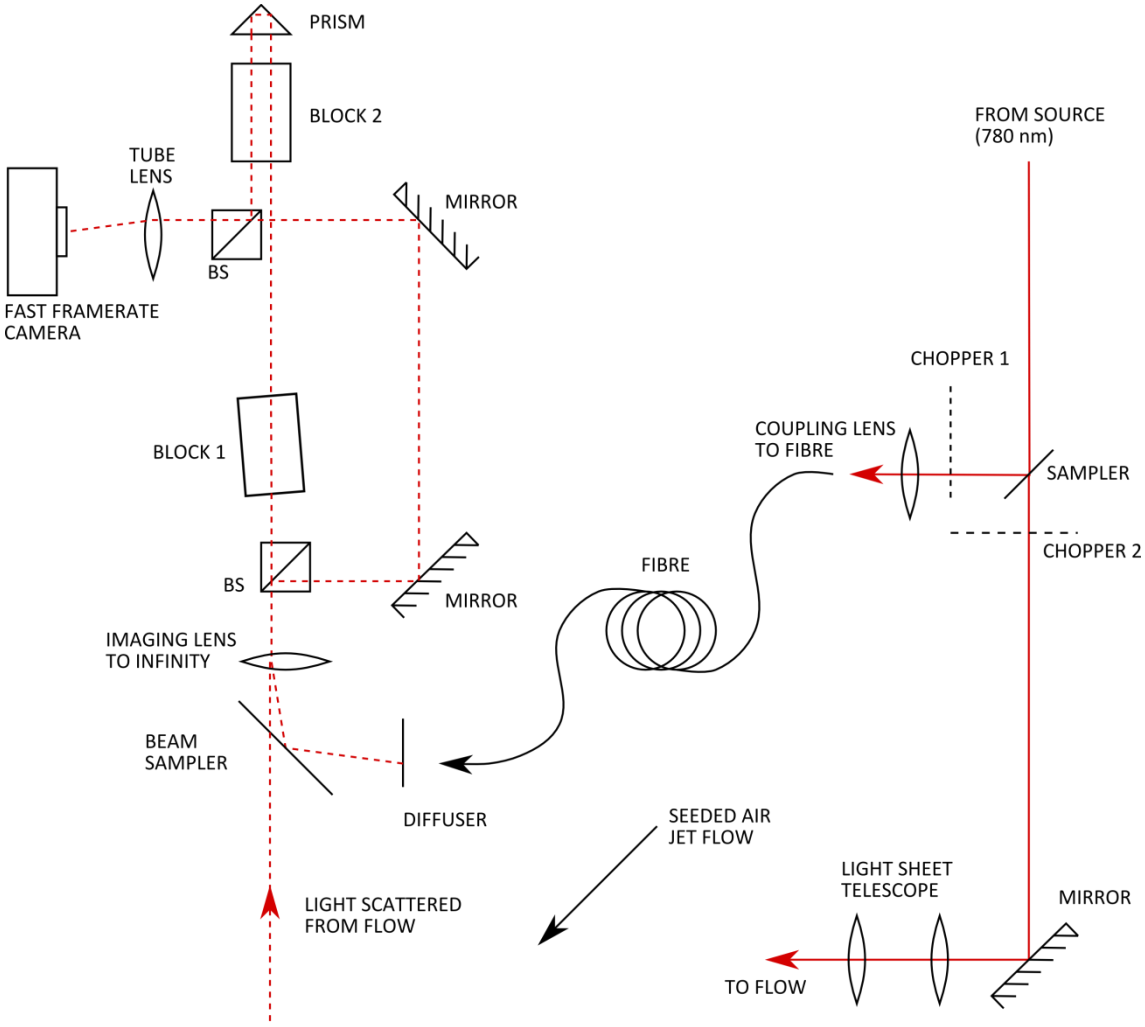


Figure 4-22: Experimental configuration for FDM I-PDV measurement of the velocity of a gas jet. This was based on the system used to measure the velocity of the disc, with the illumination configuration altered to align the sensitivity vector of the system with the velocity vector of the jet core. The reference wavefront channel is provided by the fibre delivered light scattered from the rotating diffuser. The signal channel illumination is provided by the free space light, which passed through the light sheet telescope which produced a light sheet with a constant vertical height of 50 mm.

An optical fibre was used as the source for the reference channel. The light reflected off the beam sampler was launched into a standard single-mode optical fibre (Fibercore SM750) of length 2100 mm. This was long enough to ensure that cladding modes from sub-optimal launch conditions would not propagate to the distal end of the fibre. This was important as, in order to reduce the power further than was possible through tuning polarisation at the beam sampler, the fibre launch conditions were de-optimised to that there was a mis-match between the mode-field diameter of the fibre (5.3 μm) and the projected spot size from the 20x microscope objective used

as the launch optic, which would be expected to excite cladding modes. The objective was moved away down the fibre axis from the end facet to decrease the efficiency of the launch and so reduce the intensity of light exiting the fibre. Doing this, rather than decoupling by shifting the spot laterally across the fibre core, reduced the amount of light being launched into the cladding modes. Cladding modes would have been an issue as the light would have contained a different phase from the light exiting the core due to differences in path length and quality of guiding between the core and the cladding regions. The combination of the different phases from the end of the fibre would have contributed to a phase offset in the results and local regions of phase variations from any structure in the cladding mode.

An MZI was use as the filter to convert Doppler frequency shifts to differences in intensity, aligned as described in section 4.6.2.

The air jet used was the same as that reported by Lu [61] and Charrett [68], which consisted of a smoothly tapered nozzle of diameter 20 mm through which air exited. Inside the nozzle was a number of flow straightening cells, designed to reduce turbulent currents in the jet. This air jet was capable theoretically of flow speeds of up to 94 ms^{-1}, as determined in [68]. The jet nozzle was positioned to direct the air jet along the sensitivity vector, which allowed maximisation of sensitivity to the velocity of the flow. In this configuration, the system would be most sensitive to the downstream velocity components. The air in the jet was seeded with scattering particles supplied by an industrial smoke generator (Concept ViCount Compact 1300). The smoke particles produced were produced from a light smoke oil (Concept Smoke Oil 180) which was sprayed onto a heated plate, causing it to boil and become airborne particles before emerging out of smoke generator. The particles were quoted as having diameters in the range 0.2 µm - 0.3 µm which was sufficient to produce Mie scattering from the 780 nm source that was detectable on the camera.

The air jet was contained within a chamber designed to trap the jet flow and prevent it from escaping into the wider laboratory area. There were three open panels into the chamber, one placed to allow the light-sheet into the flow, another to allow free observation of the incident region between the light-sheet and the jet, and the other panel allowed the hose into the chamber to attach to the nozzle. The nozzle was placed at one end of the chamber, directing the jet towards the other end of the chamber where an extractor fan hood was positioned to remove the seeding particles from the flow chamber and prevent them building up in the downstream region where the flow met the end wall of the chamber. The extractor fan could remove air at variable rates, and so by changing the extract rate it was possible to control the build up of seeding

particles in the chamber, allowing them to seed the mass of air in the chamber. Due to the number of open panels in the chamber design and the rate of extraction from the chamber, the mass of air in the chamber was not static but in motion, drawn into the chamber through the panels and out the extractor. The air mass moved in the same direction as the jet in the region around the nozzle and flow development region so establishing a compound flow in the measurement area. Allowing the seeding particles to build up in the chamber had the effect of seeding the background flow to allow a measurement of the velocity in the region surrounding the jet.

4.8.4 Experimental procedure

In order to gain a measurement of the flow, the following procedure was used. Firstly, any equipment that required warming up was switched on. The laser had to be switched on for at least twenty minutes to allow the Peltier cooler to stabilise the laser diode temperature. The camera was also given time to warm up, as by doing this the noise characteristics were improved. The smoke generator also had to be switched on and allowed to reach the temperature where the smoke oil could vaporise. The intake fan for the air jet was left running at a slow speed to draw any leaking smoke from the generator into the chamber and then up the extractor hood instead of allowing it to leak into the lab area while the smoke generator was idle.

During the warm-up period and between measurements, the position of the nozzle was adjusted and measured. The position was measured by noting the distance from the end of the nozzle to the point downstream at the intersection of the middle of the field of view with the centre of the jet.

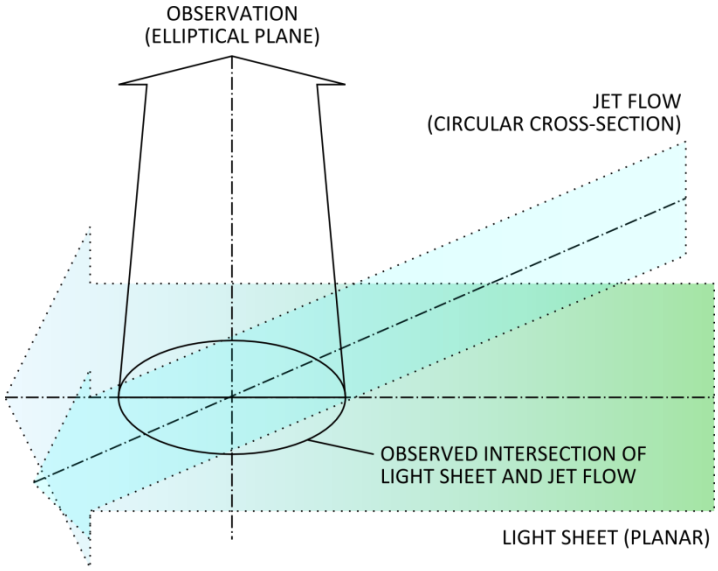


Figure 4-23: The flow measured at the intersection of the circular profile of the jet flow with the planar light sheet. This resulted in the elliptically shaped profiles of the flow observed at the camera. Arrows are drawn parallel to the plane of the bench.

After the warm up period, the beam choppers and the rotating diffuser were switched on and the air jet was put to the desired speed setting. The camera was readied to capture an image bank, but before actually capturing the images, seeding of the flow would begin to allow the volume of seed particles in the flow to stabilise. A time-series of images was then captured of the seeded flow illuminated with the chopped light at the measured distance downstream of the nozzle.

The image data was captured in an 8 bit, 256 frame time-series taken at 60 fps with a shutter time of 16386 µs. The image size was 328 x 328 pixels, which corresponded to a field of view of 35 mm x 35 mm.

The data was captured for processing using the carrier fringe method. The FFT processing kernel was a Hamming window, 200 pixels x 100 pixels, centred on the positive fundamental peak produced by the fringes in the transform. The dimensions of the kernel were selected to enhance selection of the fringes through being in the same orientation.

The air jet flow controller dial was set to the marking "60," which corresponded to a flow velocity of 73 ms^{-1} measured by a Pitot tube, and the flow was seeded. Measurements were taken at points downstream of the jet, at x/r_0 positions of 1.5, 2.4, 3.6, 4.7, 5.9, and 7.0. From the two-dimensional datasets, line profiles were generated of the profile of velocity with height through the centre of the flow, which was the short-axis of the elliptical profile received at the camera (see Figure 4-23).

The measured data was fitted to the compound flow model for the flow development region using a least-squares approach. The profiles were fitted separately to find the individual fit of the model to each of the measured profiles.

4.8.5 Calibration of the jet

The FDM-PDV measurements of velocity of the flow development region of a circular free jet as a compound flow in air were carried out on a calibrated air jet.

Measurements of the velocities in the flow development region were made using a differential Pitot tube in the high velocity central cone of the flow, shown in Figure 4-24. In the lower velocity areas, around the central cone, the velocity was measured using a hot-wire anemometer. This data was used to calibrate the velocities in the experiment and to allow the model profile to be calculated.

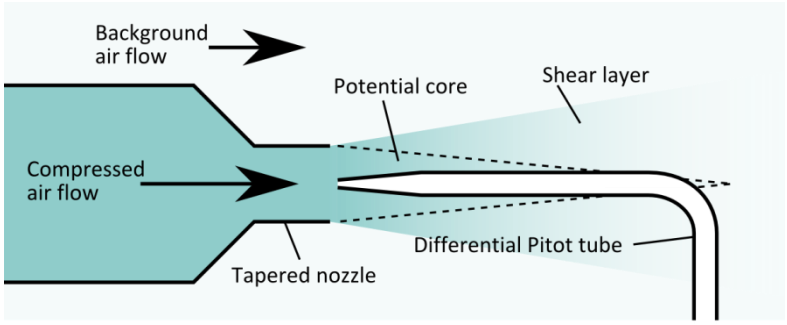


Figure 4-24: Placement of the Pitot tube within the jet to measure the pressure and hence velocity in the constant velocity core of the jet.

Figure 4-25 shows the plot of the velocity of the air jet measured at the end of the nozzle determined using the differential Pitot tube. The dial on the fan pumping air and seeding particles in the jet flow was moved between the marked settings and the pressure of the jet exiting the nozzle was measured, converted to velocity, and plotted against the markings on the dial.

The velocity of the background flow of the air around the nozzle outside the central cone, was measured using the Pitot tube, to be 4 ms^{-1}. Closer to the nozzle, air was beginning to be entrained around the edges of the jet and higher velocities were detected using a hot wire anemometer, showing velocities of 14 ms^{-1}.

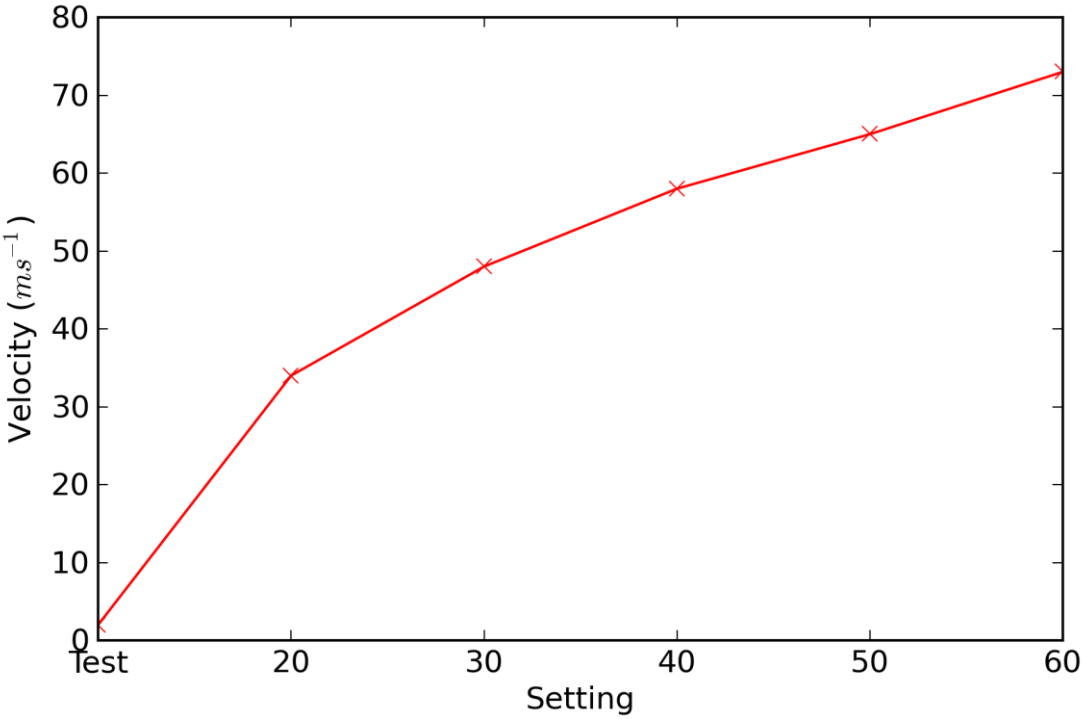


Figure 4-25: The measured velocity of the air jet at the nozzle compared with the settings marked on the fan controller. Measurements were made with a differential Pitot tube.

In the potential core, the air velocity had been determined to be 73 ms^{-1} for the measured pressure of 3160 Pa and the measured temperature of the air in the flow of 29°C. The Pitot tube was chosen as the measurement device in the high velocity, flow development region as it was specified as being capable of measuring accurately under the pressures present.

With the flow from the jet characterised, PDV data could be captured and compared to a model.

4.8.6 Results and conclusions

The results of measurement of the jet flow are depicted in the figures of this section. The cross-sectional images of the flow, formed by intersection of the flow and the light sheet, are shown in Figure 4-26. Figure 4-27 shows a three-dimensional representation of the air jet velocity as measured using PDV and fitted to the model as described in section 4.8.2. These plots are shown in more detail in Figure 4-28, one velocity profile to each axis.

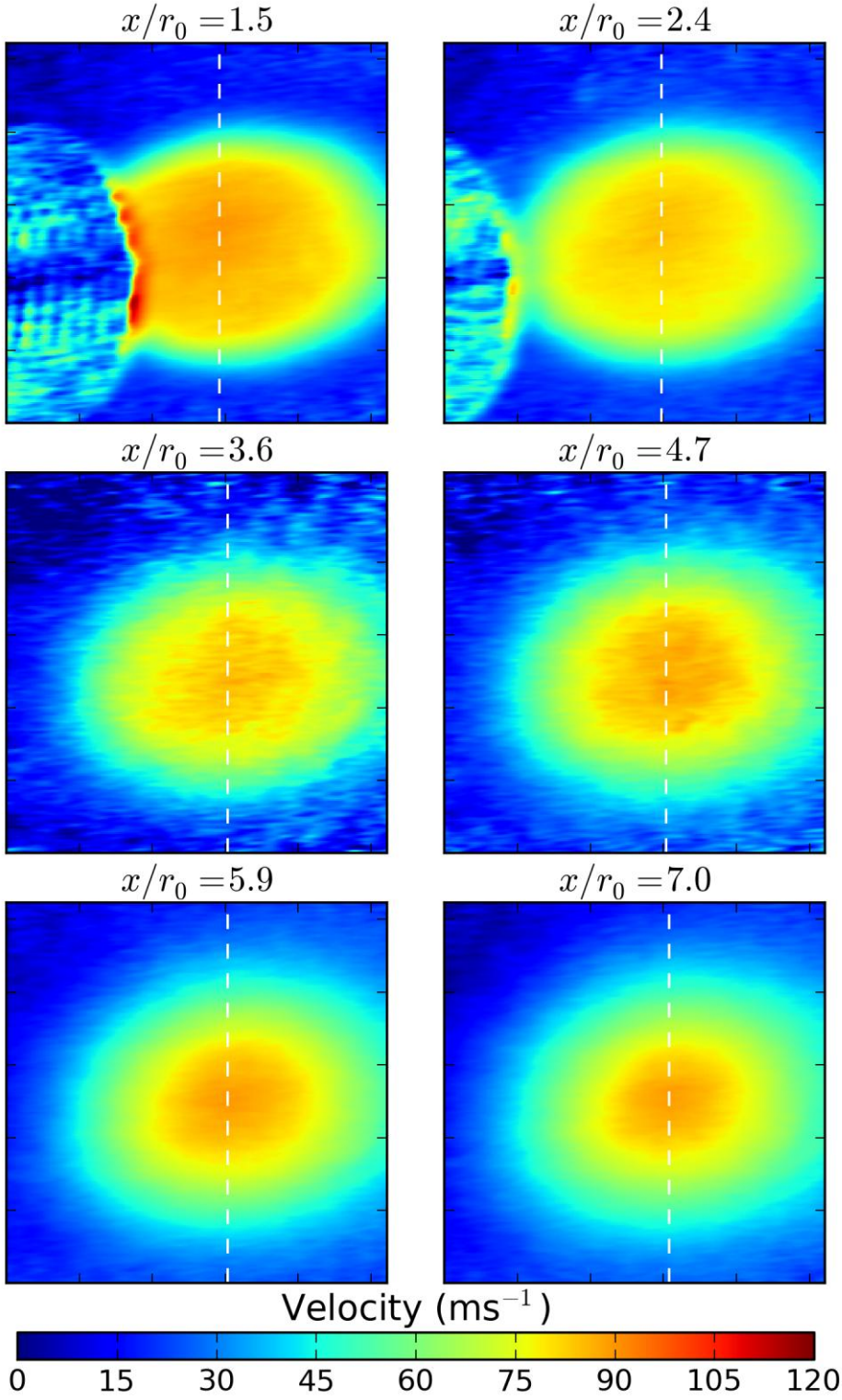


Figure 4-26: Velocity images taken through the flow. The dashed white lines indicate where the velocity cross sections were taken; the lines being on the only region in the images that would lie on a circular cross section due to the experimental geometry. Note that the slices at x/r_0 of 3.6 and 4.7 visually display more noise than the other images.

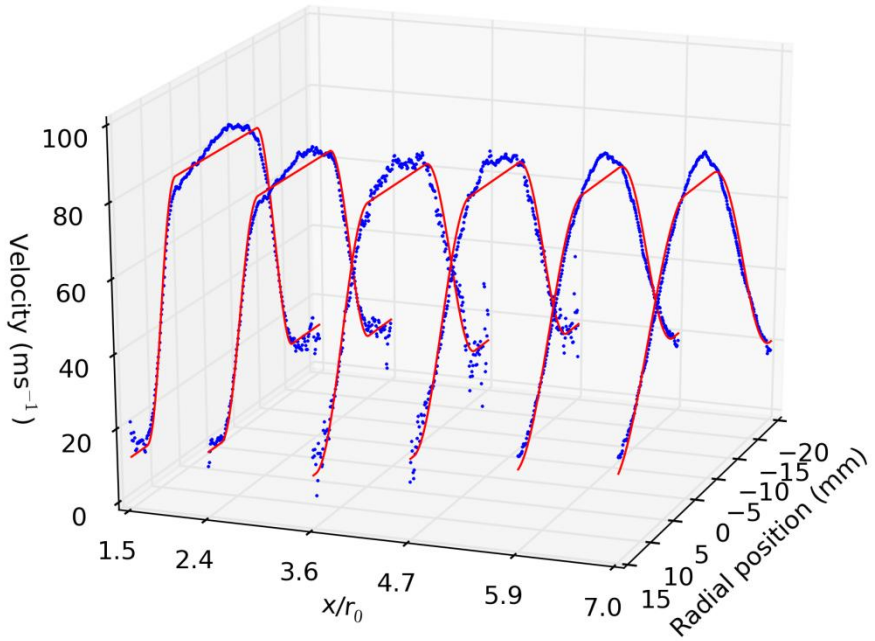


Figure 4-27: Vertical velocity profiles of the air jet flow. Profiles were taken at a number of positions (x/r_0) downstream of the nozzle though the vertical axis of the flow where the intersection of the flow and the light-sheet was on a diameter of the circular flow. The direction of travel of the jet is shown from left to right. The measured results are shown in blue and the fitted model is shown in red.

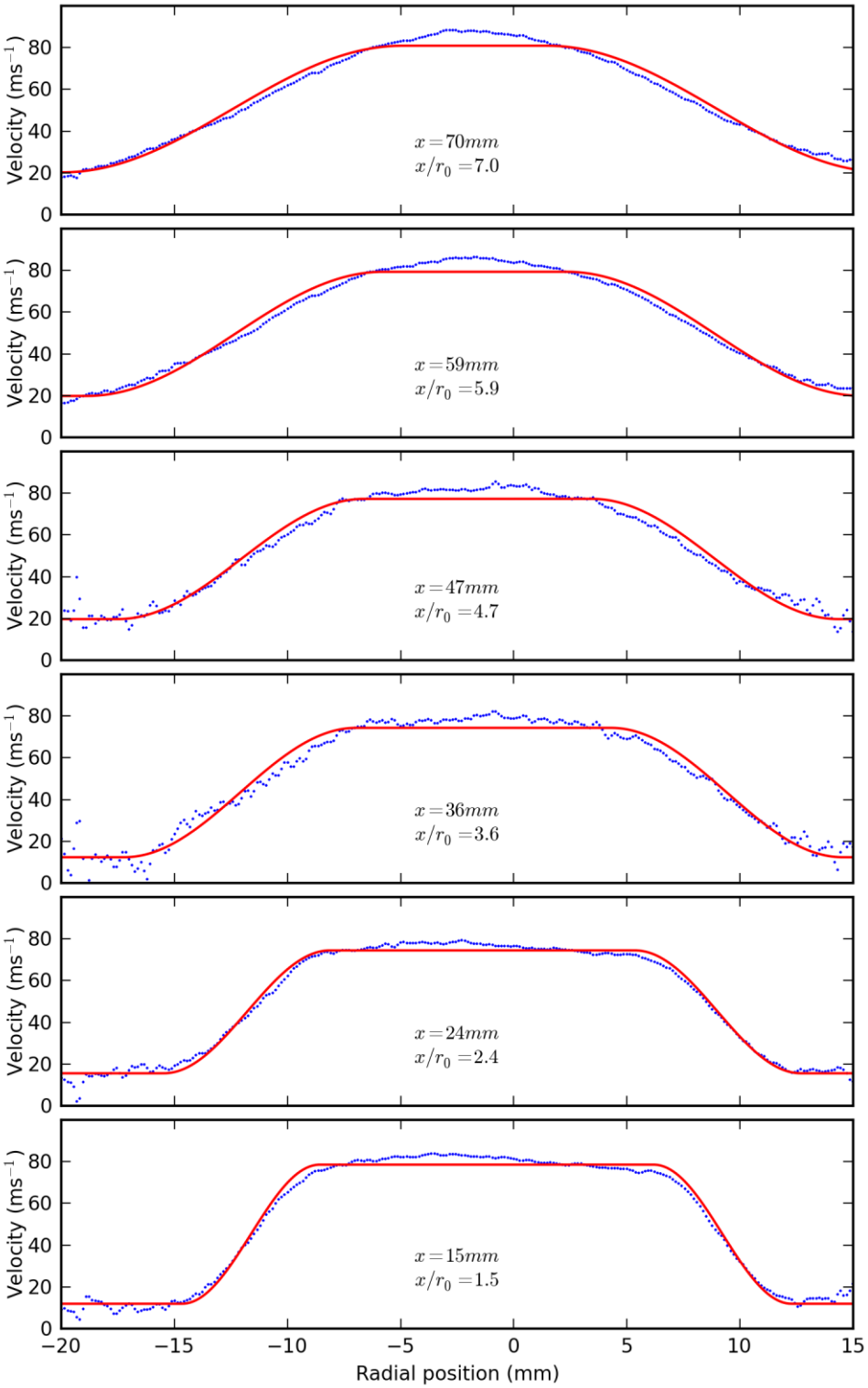


Figure 4-28: Velocity profiles of the axisymmetric compound jet taken with FDM I-PDV, shown for each measurement position, x, downstream of the nozzle. Experimental values are shown in blue, and the model is shown in red, fitted to each profile.

For a compound flow with a Λ (the ratio u_1/u_0) of approximately 0.05 for each measured result and a nozzle radius of r_0 = 10 mm, the equations of section 4.8.2 took the values shown in Table 4-1 when fitted to each of the experimental data sets.

Table 4-1: The values given by the least-squares fitting algorithm used to fit the model equations of section 4.8.2 to the experimental results, which are shown in both Figure 4-27 and Figure 4-28.

x/r_0	x (mm)	u_0 (ms^{-1})	u_1 (ms^{-1})	u bias (ms^{-1})	r offset (mm)
1.5	15	78.6	11.9	-6.5	1.2
2.4	24	74.5	15.6	-6.2	1.4
3.6	36	74.3	12.3	-5.1	1.3
4.7	47	77.3	19.7	-5.0	1.6
5.9	59	79.3	19.8	-3.3	1.7
7.0	70	81.0	20.2	-0.9	1.7

Table 4-2: Velocity difference between the jet potential core and the background flow of the fitted model in each slice, and the standard deviation of the measured profiles from the models.

x/r_0	Velocity difference (ms^{-1})	σ (ms^{-1})
1.5	67	2.9
2.4	59	2.3
3.6	62	4.1
4.7	58	3.7
5.9	60	3.2
7.0	61	3.3

The differences between the measured results and the model are shown in Table 4-2. The standard deviation of the measured results when subtracted from the model was calculated and is shown in the table, along with the velocity difference between the model core and the background flow. The results showed uncertainties (determined as

the difference between the model and the measured results) of up to approximately 2.3 ms^{-1} to 4.1 ms^{-1} between the different profiles, with the worst uncertainties on the results from the slices at positions 3.6 and 4.7 downstream of the nozzle. These two slices were those that displayed a visibly noisier velocity profile, an observation which was borne out in the error analysis.

The model of the jet was chosen to best fit the expected conditions of the experiment, though the choice of an axisymmetric air jet as a compound flow contained a number of assumptions deemed valid. These were:

- There was no turbulence inside the assembly leading to the nozzle.
- The background flow was uniform and in the same direction as the jet.
- The close proximity of the optical bench to the shear region did not influence the flow.
- The reflected flow of the jet from the downstream end of the smoke chamber was expected to contribute to the background flow without introducing turbulence or deflecting the potential core of the jet.

These assumptions sufficed as a first approximation of the jet and allowed the description of the flow to be established with a theoretical grounding. Deviations of the measured flow from the model may be due in part to the experimental conditions not quite matching the model assumptions. One possible source of error may be the presence of an unaccounted for slight vortex in the potential core, caused by imperfect flow straightening fins in the jet nozzle. This could account for the slight peak in the profiles that appears to move from positive to negative r.

As measurements were performed, the air pump would fill with smoke oil that had condensed around the fan and pooled in the bottom of the fan casing, necessitating the stripping down and cleaning of the pump after a few measurements.

The noise on the measurements at x/r_0 values of 3.6 and 4.7 has been partly accounted for by considering the time-series data, which were discovered to display a drift in the linear fringes greater than in the other measurements. This was an unexpected phenomenon, and can be explained as the visibility of the fringes being reduced due to the time averaging of the drift, which results from using the FDM algorithm used to demultiplex the data. The loss of fringe visibility can limit the accuracy of the carrier fringe method for determining phase, and it was discovered that while the fringes were of high contrast in a single frame, the demultiplexed images had far lower contrast than expected. It was also noted that the noise in the images could be removed by over filtering by choosing a narrow window when

selecting the fringe peak from the two-dimensional DFT in each demultiplexed channel. The noise levels could be increased in a measurement simply by increasing the area of the window to include more spatial frequencies. This did not account for the increased noise in the slices at 3.6 and 4.7 nozzle radii downstream however, as all the results were processed with the same parameters.

The results show an increasing r offset as x/r_0 increased. This was due to a slight misalignment of the flow to the expected flow vector introducing an error into the system which caused the centre of the flow to gradually move away from the centre of the field of view as the nozzle was withdrawn.

4.9 Conclusion

The application of FDM to I-PDV allowed the successful multiplexing of the reference and signal channels. This was demonstrated on two velocity measurement configurations: firstly the velocity of the surface of a disc, and secondly from particles suspended in an axisymmetric air jet.

I-PDV results obtained using the carrier fringe method are highly susceptible to the shape of the window function. This can lead to uncertainty in the measurements.

FDM I-PDV suppresses large errors produced by drifting fringes overwhelming the Doppler shift signal. The FDM technique is capable of suppressing the majority of the effect of drifts seen in the work of this section. This has the advantage of inherently protecting the system from drift induced error, due to the fact that the reference and signal images are taken in parallel, and so these errors present in the system are present in all channels and consequently are able to cancel. It was possible to further reduce the residual error from drifting fringes by reducing the number of frames in the image time-series. This reduced the total drift seen in the time period and so allowed for a greater fringe contrast after demultiplexing,

4.10 Future work

The stability of the system may in future be improved through a number of means. The MZI has two possible output windows from which images may be obtained which can be used to normalise the fringe images [61]. As the outputs of the two windows are π out of phase, it would be possible to impose the two images on a single camera and have the fringes of one image lie in the dark regions between the fringes of the other

image, and by using FDM be able to separate the two output images from one another on top of the FDM signals being received from each channel.

Active compensation may be employed also by tuning the laser source to a resonant absorption feature in a rubidium vapour cell to detect drifts in the laser frequency or a secondary interferometer may be created from the elements of the MZI to test for drift in the positions of the optical elements which could induce an error. Any detected drift by either of these methods could be compensated for by either driving a mirror on a piezo-electric mount or tuning the laser current to restore stability.

Removing the mechanical modulation sources in each channel would improve the stability of the FDM algorithm, as jitter and chirp in the modulation broadens the peaks in the DFT, necessitating the use of wider window functions to accommodate this effect. Shaped windows and the positions of the peaks within these windows can affect the outcome of the algorithm. Improving the resolution of the peaks would help ensure broadening effects are minimised. Imposing a sinusoidal modulation via drive current modulations are an option to improve the system and removes the need for mechanical elements. An additional benefit of sinusoidal modulations are that in an appropriate sample, the peak obtained from such modulation appears at a single frequency, and so no higher harmonic information need be present in the DFT spectrum, and so when windowing the peaks during FDM demultiplexing, the higher harmonic data need not be lost to the window function, and the higher harmonic peaks need not clutter up the spectrum. This would open up the system to a greater number of channels, or allow for greater freedom in locating channels in frequency space.

The system was demonstrated using two channels: one for the measurement signal and the other for the reference wavefront. Future work would also have to include the scaling up of the number of channels to facilitate three-dimensional measurements of velocity or an increased number of measurement planes in the flow volume.

Adapting the FDM I-PDV technique to dynamically moving flows would be interesting, especially to those with a periodic component in time, as the FDM technique is akin to lock-in methods, it may be possible to use it in measuring specific turbulences or mixing flows which can be selected as a frequency in the image time-series.

5 Error analysis of the FDM technique

5.1 Introduction

In this chapter, the FDM algorithm will be further examined. The propagation of noise through to the final result will be examined as a function of both the characteristics of the noise and of the characteristics of the filtering applied to the time-series. The influence of noise will be considered on simulated data and on data captured from the camera module used in the body of this work. The effect of time-series length on the resolvability of peaks will also be covered, along with the effect of the shape of the modulation signal. The discussion will compare FDM processing to an equivalent technique of time averaging, a technique often used to suppress noise in a system but which is not capable of channel multiplexing, to place the results in context.

5.2 The Fourier domain

The FDM technique involves Fourier transforming the data in a time-series and applying Parseval's theorem to convert the frequency response to an intensity, as described in chapter 2. The peaks produced by a channel in frequency space are masked and processed to obtain the intensity of the image in that channel. It is necessary, therefore, to be able to produce a strong response of narrow, sharp peaks in the Fourier transform in order be able to mask them effectively. One method that may be used to produce a sharper peak is to record many samples of that signal over time, as the frequency range in a DFT is split into a number of equal sized bins, the number of which are determined by the number of samples captured.

In this section, simulated time-series were produced containing two representative periodic signals, of frequency 7 Hz and 17 Hz respectively so to avoid situating one signal atop the harmonics of the other, and their Fourier spectra were plotted. The simulations assumed a 60 fps sampling rate, as used in the FDM I-PDV experimental work, and a normalised signal in each channel, ranging between zero and one arbitrary units of intensity. Figure 5-1 shows the case of two channels with a sinusoidal form sampled over a range from 9 to 1024 samples. It can be seen that, as the number of samples increases, the intensity of the peak also increases, and from theory this would be expected as the area under the peak should remain constant. Peaks are discernable all the way down to 90 samples, which corresponds to approximately ten cycles of the slowest modulation channel (7 Hz), though at a greatly reduced intensity. When nine samples are used, corresponding to one cycle of the 7 Hz

channel, no peaks are discernable in the Fourier spectrum. This would suggest that a minimum of ten cycles of the slowest modulating channel should be captured. However, in order to improve the appearance of peaks when there is noise in the time-series, it will be necessary to capture more samples to raise the peaks above the background.

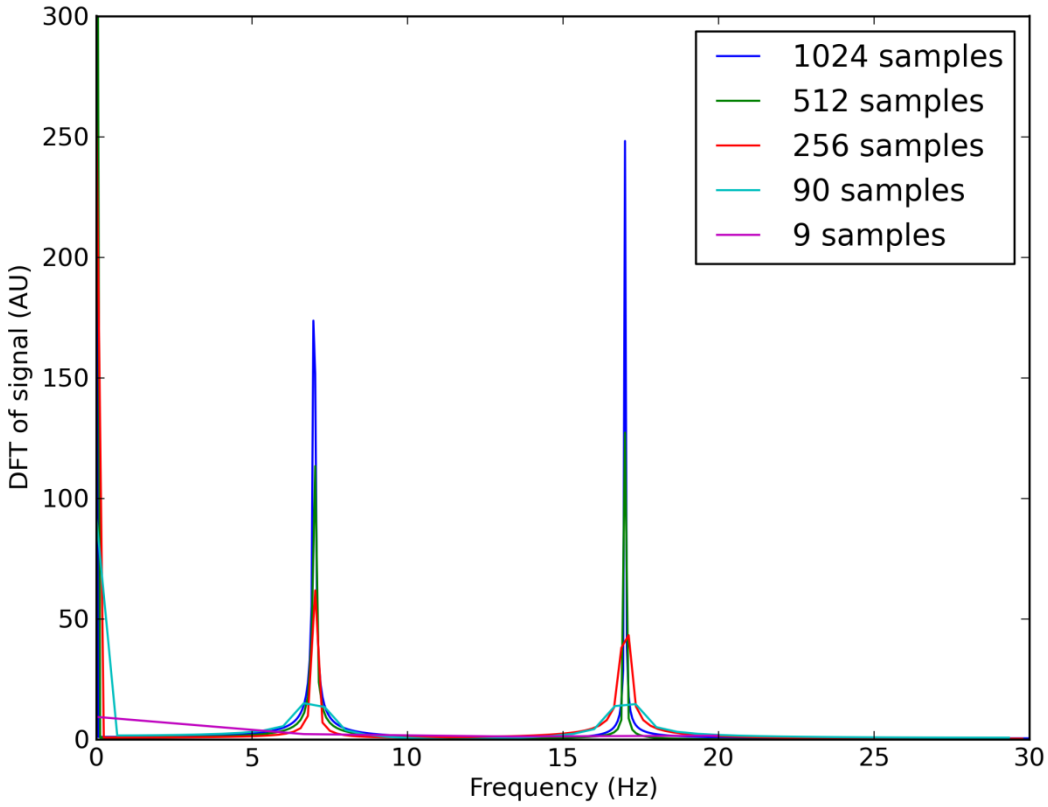


Figure 5-1: Frequency spectrum of two simulated sinusoidal waves at 7 Hz and 17 Hz from different sample length time-series.

Figure 5-2 shows the same simulation performed using square waves. The square waves produce a significant number of additional peaks in the spectrum. Again, sampling ten cycles of the slowest modulation produces discernable peaks, though as there is increased harmonic content from the two channels, the peak intensity is reduced as energy has been transferred from it into the harmonic peaks characteristic of the square wave modulation.

It can be seen from this that a smoothly varying signal is preferred to a sharp edged signal like a square wave, as these require additional frequencies to realise their edges. Though the FDM algorithm has been proven to function despite the increased

harmonics which come with the use of such a modulation, it is possible to achieve a cleaner spectrum of reduced harmonic content. A cleaner spectrum would allow the multiplexing of more channels and produce less cross-talk. It can also be seen that, as the peak sharpness decreases, the width of the peaks increases. Thus, when fewer samples are recorded, a broader window function should be used in order to obtain the values from the peaks. This takes up more space in the spectrum, and reduces the number of possible channels.

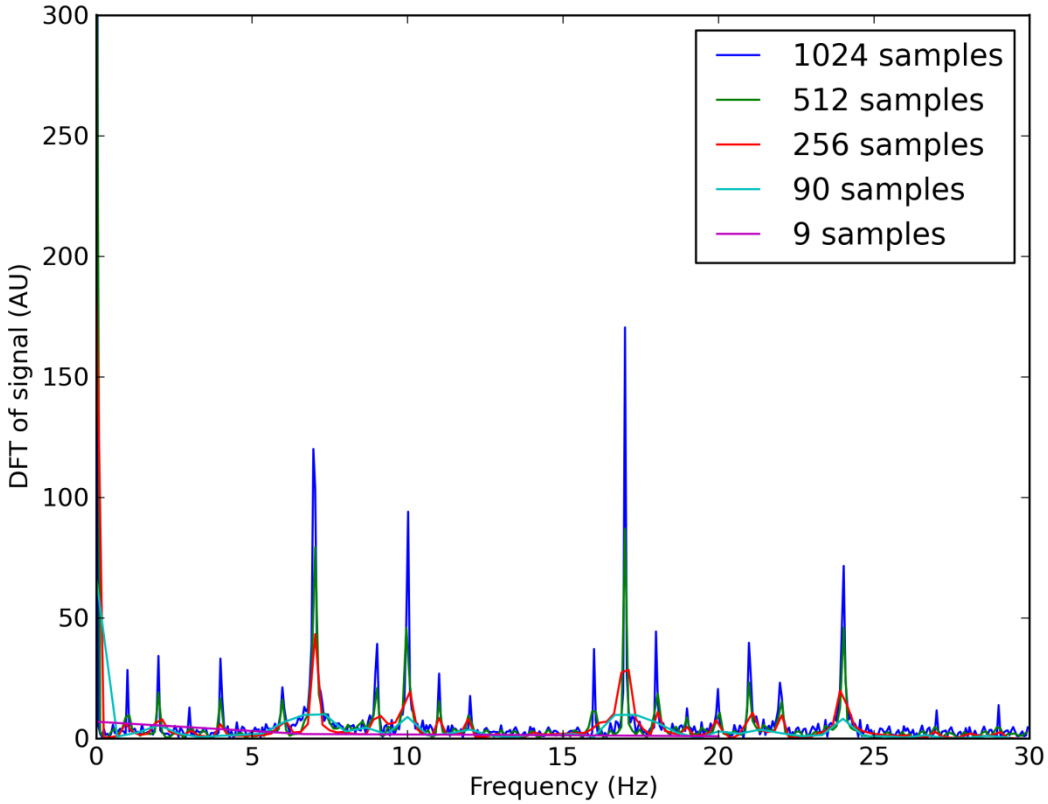


Figure 5-2: Frequency spectrum of two simulated square waves (50% duty cycle) at 7 Hz and 17 Hz from different sample length time-series.

5.3 Image reconstruction

5.3.1 Phase error effects on image reconstruction of a fringe pattern

An unknown factor in the use of the FDM algorithm was the influence on the image reconstruction of any changes in the scene being imaged during the time-series acquisition.

To investigate the influence of temporal effects on the FDM image reconstruction, a simple one-dimensional sinusoidal fringe pattern was simulated and was put into a time-series with a constant phase offset between successive time samples. A square wave modulation was imposed on the time-series along the time axis to allow the application of FDM to reconstructing the fringes. Figure 5-3 shows the entire time-series with a large phase drift of 4π radians between the start and end of the time-series. The simulations were repeated to obtain measurements for phase drifts in a range from -2π to 2π, though in practice, the drifts experienced were of the order of two radians in the time period, giving a drift rate of around 0.5 rad/s. Fringe reconstruction was attempted for the techniques of FDM and time averaging.

A sample of the reconstructed fringes is shown in Figure 5-4 for three phase drifts. It can be seen from this figure that the reconstructed fringes from both techniques of FDM and time averaging contain a phase error. The time averaged fringes have maintained a sinusoidal profile, but those obtained from FDM can be seen to have developed a higher order distortion in addition to the phase offset. As the techniques of FDM and time averaging sum along the time-series, the phase offset error in the reconstructed fringes can be explained as an analogue of the effect of integrating a sinusoid. A sinusoid which is summed over infinity will equal zero, and the same result is obtained when summed over an integer multiple of 2π phase. With increasing phase drift, the time-series of a given point on the fringe pattern approaches a sinusoidal form and, when this is summed, any whole number of periods in the time-series have a net zero effect on the phase, but the fractional period left does not cancel out and becomes the phase offset on the point. It was found that a direct relationship existed between the phase drift and the reconstructed fringe phase, where a 2π phase drift in the time-series resulted in a π phase error in the reconstruction.

Another interesting effect of a phase drift in the time-series is that it affects negatively the visibility of the reconstructed fringes. Figure 5-5 shows how the visibility of the reconstructed fringes varies with the phase drift. With no drift present, the fringes maintain a visibility of unity for both time averaging and FDM. As the magnitude of the fringe drift increases, the visibility decreases, reaching zero around 2π radians of drift. This corresponds again to the analogy of summing of sinusoids discussed. It is worth pointing out that the FDM result reaches its minimum value later than the time averaged result, and this is due to the higher order phase distortion seen in the phase reconstruction, causing the summation not to reach zero at exactly 2π.

Fringe drift should then be controlled in full field interferometry when accurate measures of phase in the image is required in any time resolved measurement.

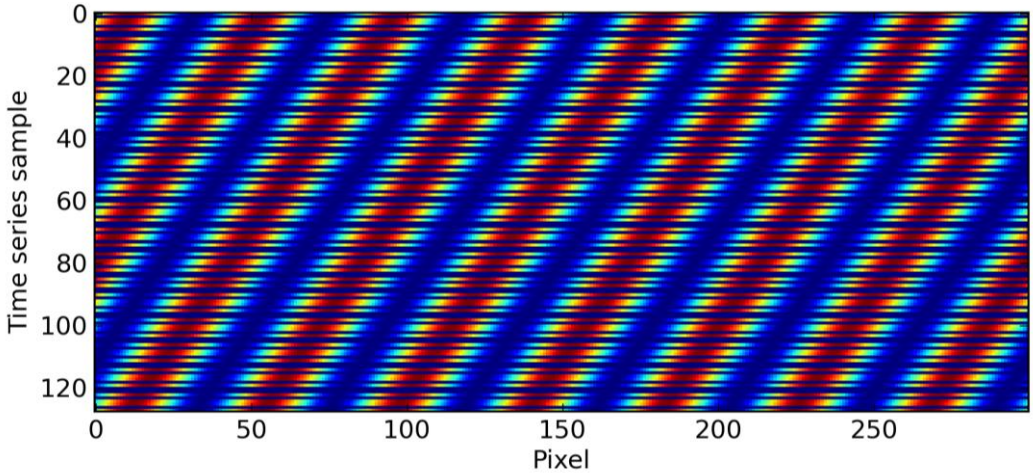


Figure 5-3: Time-series of a simulated one-dimensional fringe pattern with a phase drift in time under square wave modulation.

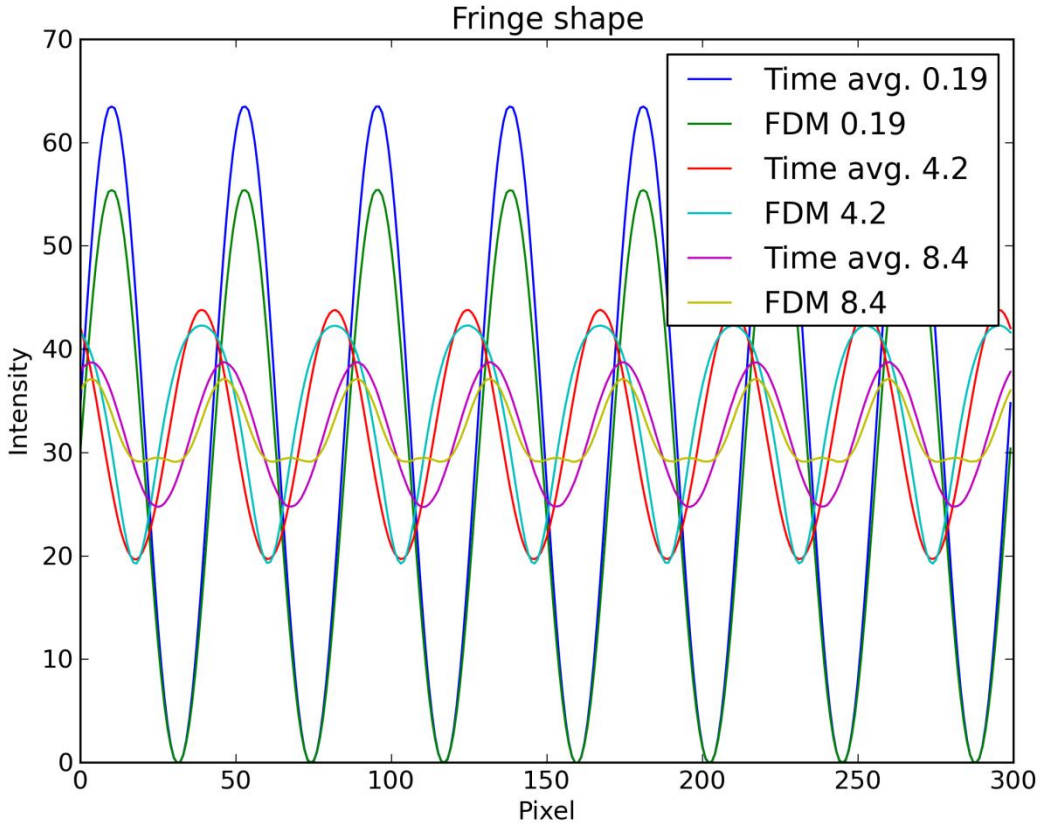


Figure 5-4: A resulting fringe pattern when a known phase drift is present in a time-series. Fringes resulting from processing via FDM and time averaging (Time avg.) are shown for three phase drifts of 0.19 rad, 4.2 rad and 8.4 rad.

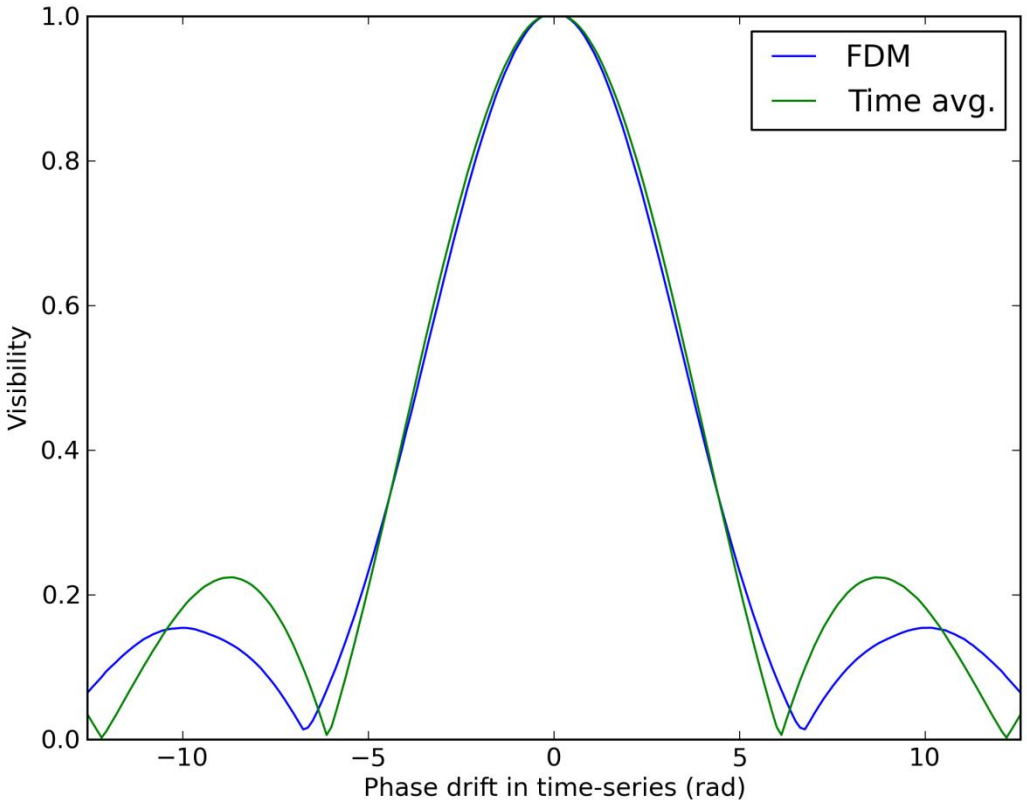


Figure 5-5: Plot of visibility of the fringes with total phase drift in the time-series.

5.3.2 Low intensity image reconstruction

While it was clear that the FDM algorithm was able to reconstruct images with reduced noise, a question remained as to how well the technique was able to perform when the illumination intensity was low, reducing the signal to noise ratio.

To examine this, an experiment was conducted to reconstruct an image at different intensities. A fringe pattern was imaged onto the Baumer camera, used during the research reported in this thesis, and image time-series were captured. The time-series were square-wave amplitude modulated in time. The peak intensity of the light source was varied to reduce the maximum number of counts on the camera at fringe maxima. Figure 5-6, which contains the reconstructed fringe patterns from the FDM demodulated time-series, shows that the fringes are well reconstructed from even low counts at fringe maxima in individual frames of the times series. The fringe pattern with 64 counts at the maxima was found when reconstructed to have the equivalent noise characteristics of a single frame capture of the fringe pattern on the same

camera with approximately 450 counts maximum intensity. This number of counts is, of course, impossible to achieve on the Baumer sensor in 8 bit mode as it would saturate at 255 counts, though fractional levels can be obtained with the application of an averaging technique to a time-series. When the peak intensity in a single frame of the time-series was reduced to 8 counts on the camera, the FDM result was found to be equivalent to a single frame capture as if the fringes in it had a peak intensity of 64 counts. Using time averaging instead of FDM on the equivalent time-series lengths would yield equivalent single-frame intensities which were 10% more than those of FDM.

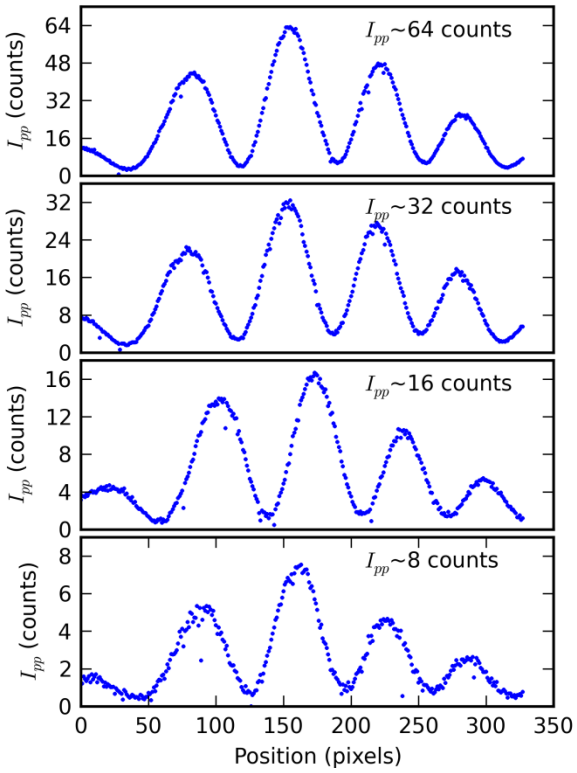


Figure 5-6: Experimental application of the FDM algorithm to image reconstruction at low intensity. Shown here are four plots of a section through a reconstructed linear fringe pattern captured in a time-series when the peak intensity at the centre of the pattern was 8, 16, 32 and 64 counts respectively. There are a few points with low I_{pp} which were caused by banding noise of vertical stripes in the images.

This demonstrates that the FDM algorithm is highly tolerant of low intensity signals, effectively improving the signal to noise and producing good quality reconstructions of low intensity images.

5.4 Noise propagation

5.4.1 Noise distribution in the power spectrum

The noise in a time-series will have a representation in the Fourier domain. This has implications for the FDM algorithm, as a proportion of the noise will be captured by the window functions and carried through to the demultiplexed result. A Fourier spectrum of noise is shown in Figure 5-7 for 256 values of Poisson (counting noise) and Gaussian (electronic noise) distributed noise arrays. The resulting spectra show that the noise is evenly spread throughout the spectrum. A discussion on the implications of noise in the time-series is provided in the following text.

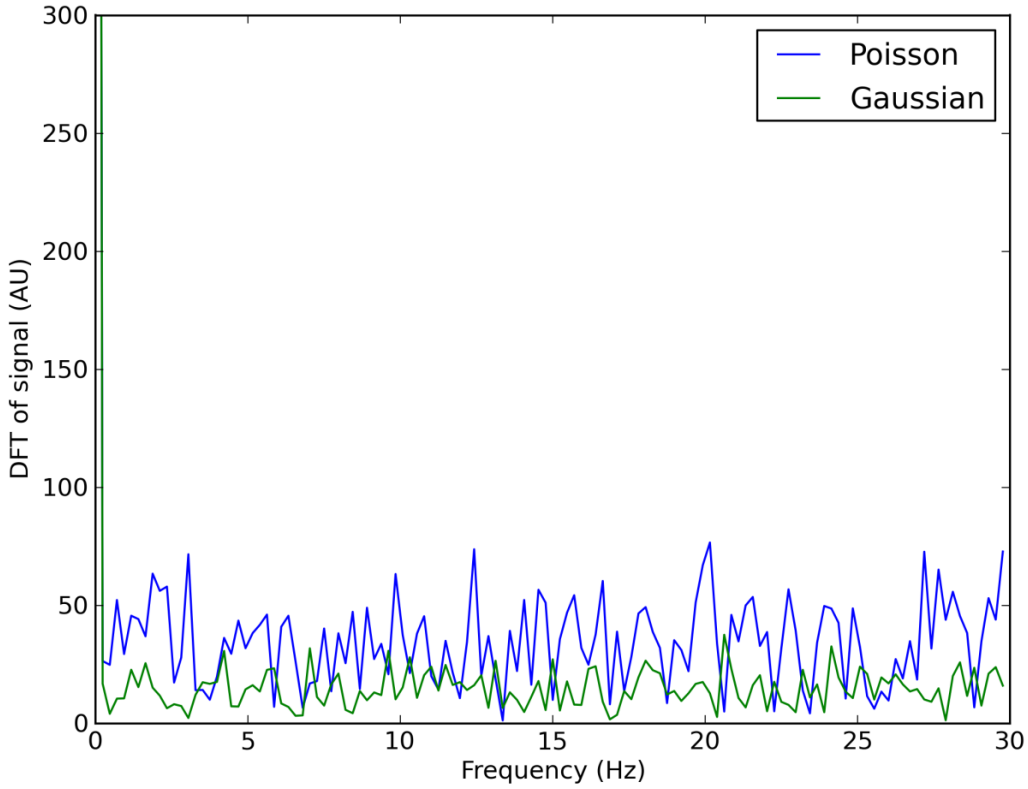


Figure 5-7: Fourier transform of two time-series containing only Poisson and Gaussian noise. The noise is distributed throughout the spectrum.

The comparisons made here should be used with care, as the measurements noise of the FDM algorithm discards the DC background level and maintains a filtered proportion of the standard deviation of the noise about the DC background, whereas the measurements using time averaging maintain the DC level and suppress the standard deviation. These distinctions should become clear by the end of this section.

5.4.2 Window width on noise

The impact of noise in the DFT that was present in the pass region of the window function on the reconstructed result was investigated in simulation by making measurements of noisy time-series when window functions of various widths were applied in the FDM algorithm.

A model was developed to simulate the effect of applying the FDM algorithm to a time-series consisting of Gaussian noise (using a mean of 4 levels and a width of 2 levels), the dominant form of noise generated in the camera. The time-series was 256 frames long and assumed that only a single simulated pixel was required to obtain a relevant sample. The demultiplexing was applied assuming a sampling rate of 60 fps. A rectangular frequency selection window function was created and the width was increased with each measurement cycle. Three centre frequencies were investigated, meaning three window functions were created, centred respectively towards the low frequency edge, centre of the Fourier domain, and toward the high frequency edge at 7 Hz, 15 Hz, and 25 Hz, and the widths of these were each increased at the same rate. The widths ranged from 0 Hz to 60 Hz, which took the window widths to twice the length of the frequency domain. This allowed the simulation to include edge effects of window function length overflow in the FDM algorithm. For each width of the window, the simulation was run 100 times and the results were analysed to obtain the mean and standard deviation of the results at a given window width.

The plot in Figure 5-8 shows that increasing the width of the window in FDM passes increasing noise to the final result. This can be explained by the fact that, as the width of the window increases, it samples more frequency bins, and this effectively means that, by applying Parseval's theorem, more values are summed into the final result, thus increasing the contribution to the final intensity. However, in Figure 5-8 the gradient is observed to be discontinuous when the mean intensity is plotted against the expected frequency bandwidth of the windows. When plotting against the actual number of frequency bins that are present in the window functions, which is shown in Figure 5-9, the results are observed to follow the same continuous curve where the intensity is proportional to the square root of the number of frequency bins in the window function. This is an effect of the discrete nature of DFT functions, which bin frequencies, and the frequency selection method employed in creating the window functions. The window function selects frequencies by including bins which fall inside the specified range. On reaching the end of the frequency bin array in memory, the algorithm truncates the window at the highest absolute value, rather than employing wrapping or mirroring as sometimes employed in computational implementations of

DFT based algorithms. The reasons for this are to ensure the window function array length does not exceed the length of the time-series under analysis and stray out of valid memory when passed to the custom C library, and also to avoid duplicating bins in any padding or reflection routine, as the final result is highly sensitive to the presence of peaks due to the square term in Parseval's theorem. This accounts for the discontinuous nature of Figure 5-8 which is plotted against the desired bandwidth and not the actual bandwidth of the discrete window function which truncates at the edges of the frequency array, causing the discontinuities present in the plot. When plotting for the actual number of bins included in the window function, as shown in Figure 5-9, these discontinuities are not present. A clustering of results can be seen as the number of bins increases, this being due to the results which would have become discontinuous being slowed in increasing intensity by the ceasing of window expansion at one edge.

Another issue that the discrete nature brings is that selecting a bandwidth becomes unreliable, as there may be regions of the spectrum where the window function includes additional bins compared to other channels and thus becomes wider, or may be truncated at the edges, changing the noise characteristics between channels.

It can be further seen that the relationship between the number of bins and the expected frequency bandwidth of the windows follows the trends shown in Figure 5-10, which shows the window width increasing with frequency until a change in the linear relationship that occurs when the gradient relationship changes discontinuously from $I \propto m$ to $I \propto m/2$, where m is the gradient of the slope. The change in the gradient relationship is a result of the window function reaching the frequency domain limits on one side and so halving the number of bins being added with each increment in expected window bandwidth. The filter width at which the gradient change is observed will increase as the centre frequency is increased. This plot is important in gaining an understanding of the effect of the window in the FDM algorithm.

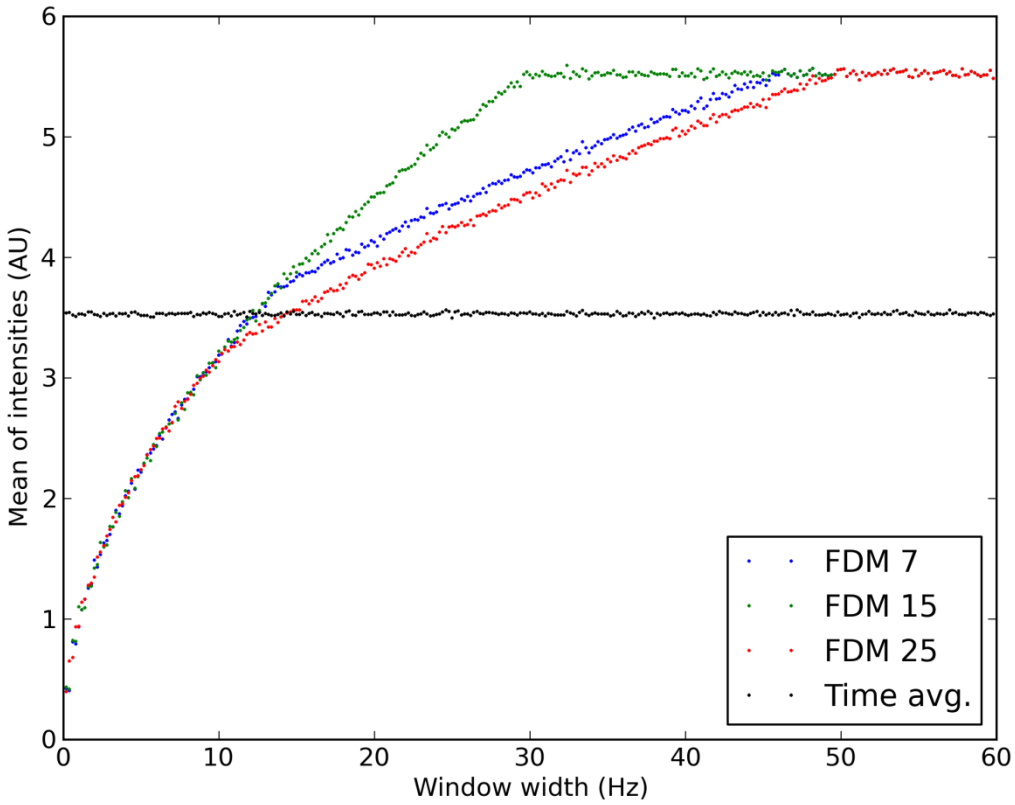


Figure 5-8: Plot of the effect of increasing the width of the window used in demultiplexing, for three window centre frequencies of 7 Hz, 15 Hz, and 25 Hz (labelled FDM 7, FDM 15 and FDM 25 respectively). The same time-series processed as time averaging (labelled Time avg.) is also shown as comparison.

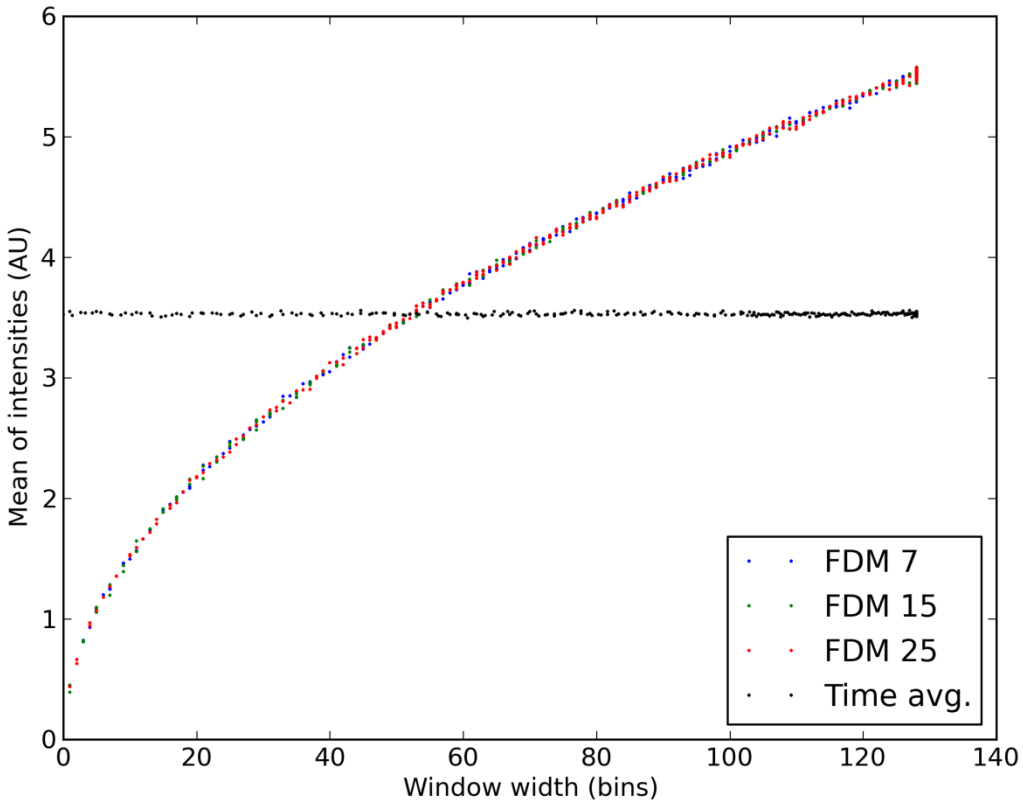


Figure 5-9: The same results as plotted in Figure 5-8, but plotted against the actual number of windows used in the calculations. The relationship between the FDM results becomes clearer, and also quantisation effects become apparent in the frequency axis, visible as vertical stacking of the points.

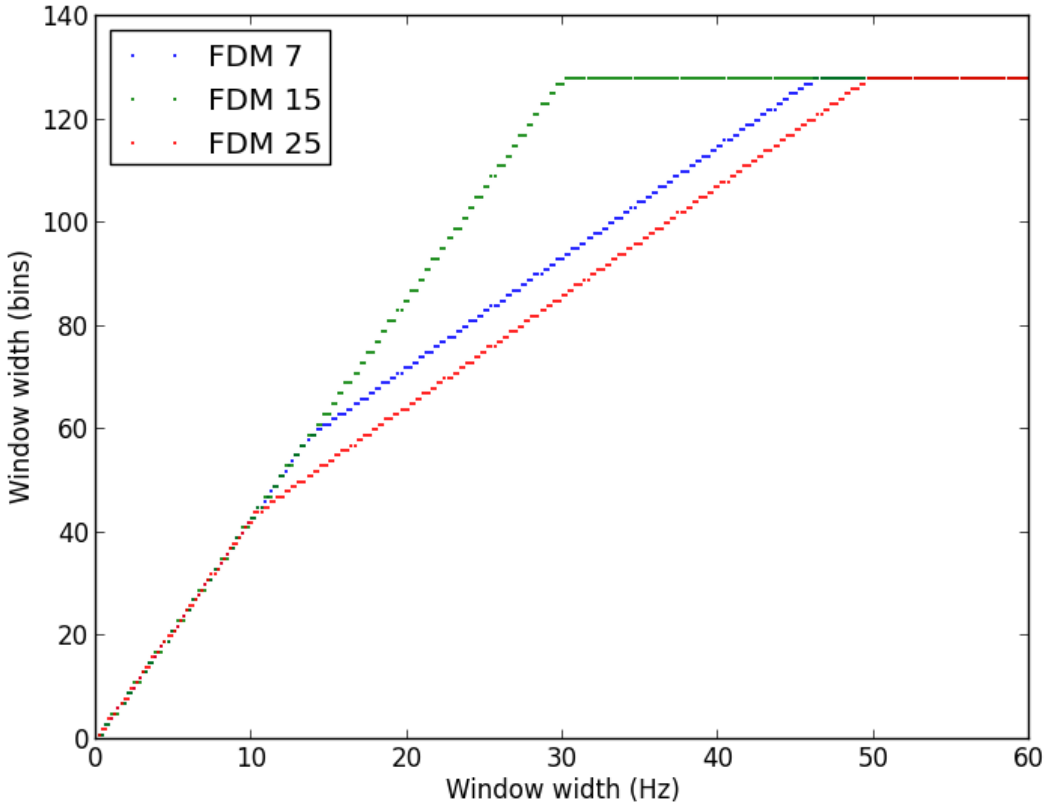


Figure 5-10: The variation of the actual number of frequency bins included from the DFT when a width in Hz is selected. This plot shows that the number of bins included varies linearly with frequency as expected, but a change in this gradient is observed when the edge of the DFT is reached and the window would exceed the Nyquist limits of the DFT. The results are shown for the cases of windows centred at 7 Hz, 15 Hz, and 25 Hz (FDM 7, FDM 15, and FDM 25 respectively).

5.4.3 Effect of window position on the noise

The effect of increasing the window centre frequency on the noise admitted was investigated.

A simulation was developed to produce a time-series for a single pixel consisting solely of Gaussian noise. This noise had a mean of 10 counts with a standard deviation of 2 counts, these values selected as a conservative match to the conditions in the Baumer camera. This was passed to the demultiplexing algorithm along with a window. The demultiplexed result consisted of a single value of intensity for the simulated pixel. This was repeated 100 times for a given window centre frequency before the window

was centred on another frequency and the process repeated. The time-series was also processed using time averaging, taking the mean of the entire time-series. The simulation assumed that the pixel recorded 256 values in the time-series at 60 fps. The windows used had a width of 1.5 Hz, which corresponded to the inclusion of seven frequency bins in the calculation. The number of bins was fixed in the calculations to avoid biasing the results by varying numbers of bins between windows at the centre frequencies.

In this way, a mean (Figure 5-11) of 100 intensity values was obtained at each frequency up to the Nyquist limit for the FDM and the time-averaging approaches. It can be seen that the background noise does not vary with the centre frequency of the demultiplexing algorithm window. The background noise in the FDM results is reduced by more than a factor of seven compared to the level of noise that remains after time-averaging, though this of course is due to the selection of the DC level used in the analysis, and Figure 5-17 shows how this offset from FDM can be varied. The results as presented here show the comparison between the final results of using two approaches to obtaining an image from a time-series, but most likely in practice the DC offset would be subtracted.

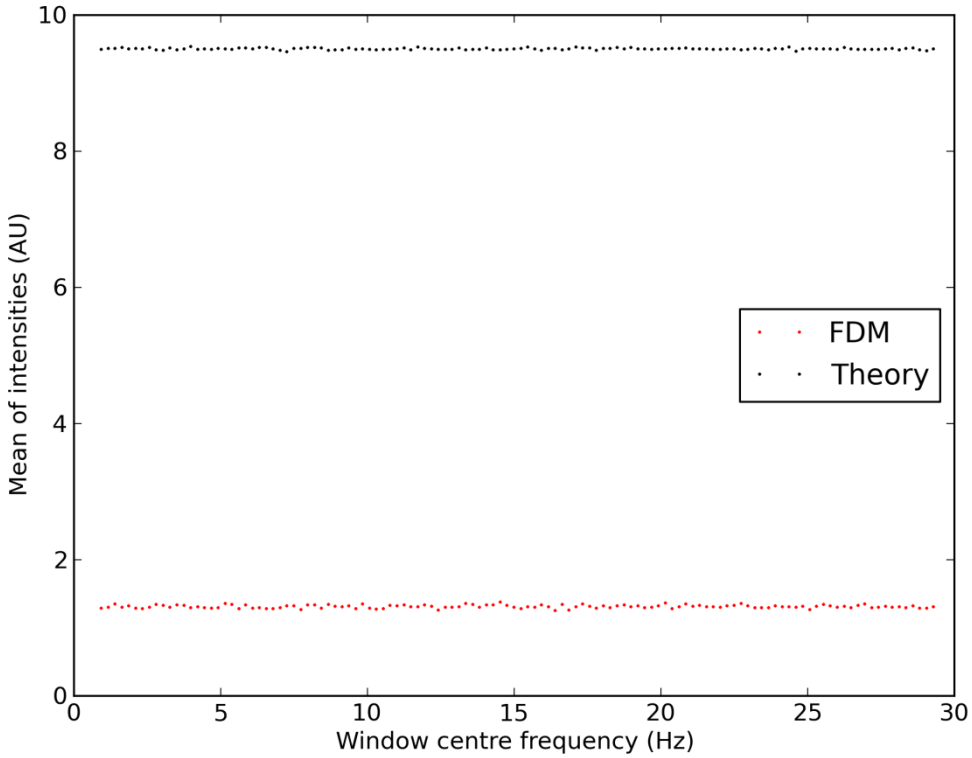


Figure 5-11: Mean intensity with increasing window centre frequency in the FDM algorithm. The results are derived from 100 repeated calculations on simulated time-series of 256 frames length. Values were determined applying the FDM algorithm with a 1.5 Hz wide window at a given frequency (FDM), and, for comparison between possible approaches, as time-averaging determined as the mean of the entire time-series without DC subtraction (Theory). The simulation assumed a 60 fps capture rate, with Gaussian noise in the time-series of intensity 10 counts and a spread of 2 counts.

5.5 The Baumer HXC13 camera

The camera used in all of the practical measurements was the Baumer HXC13 camera. It was based on the LUPA 1300-2 CMOS sensor from Cypress Semiconductor Corporation, which was quoted as being capable of 500 frames per second at full SXGA resolution (1280 pixels by 1024 pixels) though in practise the camera was capable only of 430 fps unless using experimental drivers. By defining a region of interest, it was possible to increase the frame rate of the camera, as each pixel in a CMOS image sensor is an independent sensor and so can be addressed separately. This is in contrast to the other common camera sensor type, CCD, on which the pixels are read out in rows, making the use of sub-regions on these sensors less efficient.

As each pixel on the Baumer camera is a separate sensor, it was possible to make a noise map of the sensor. This was achieved by taking images when no light was able to reach the image sensor due to the lens cap being on, i.e. the dark signal. Images were recorded to provide information on the noise that would be present when capturing a single frame, and also the noise that would be present when making measurements using an average of multiple frames (time averaged) and with application of the FDM algorithm. Figure 5-12 shows the background DC level on the sensor remaining when the images in the time-series are averaged along the time axis to produce an average value in time for each pixel. Figure 5-13 shows the noise on the sensor after application of the FDM algorithm, effectively the standard deviation of the background noise. An image has not been shown here for the noise on the single frame as it is visually identical to the image produced from time averaging. The mean (typical pixel) value of the dark signal in a single frame capture, taken from 100 measurements, is 2.3 counts, while from 100 time averaged measurements from 100 frame time-series, the typical pixel value is also 2.3 counts. The typical pixel value in an FDM image with a 10 Hz window width and averaged from 100 measurements is only 0.073 counts. Variation between single frame measurements were given by a standard deviation which was found to be 1.2 counts, while time averaged results had less variation between measurements, at only 1.1 counts. Again, the FDM measurements were more consistent, with the inter-measurement standard deviation at 0.029 counts in 100 measurements. This is summarised in Table 5-1. Figure 5-14 shows the effect of increasing the integration time on the number of counts the camera detects.

Table 5-1: The mean dark signal value of a typical pixel and the standard deviation of this between measurements when using the measurement methods with the Baumer HXC13 camera.

Method	Typical pixel signal level (counts)	Measurement standard deviation (counts)
Single frame	2.3	1.2
Time averaging	2.3	1.1
FDM	0.073	0.029

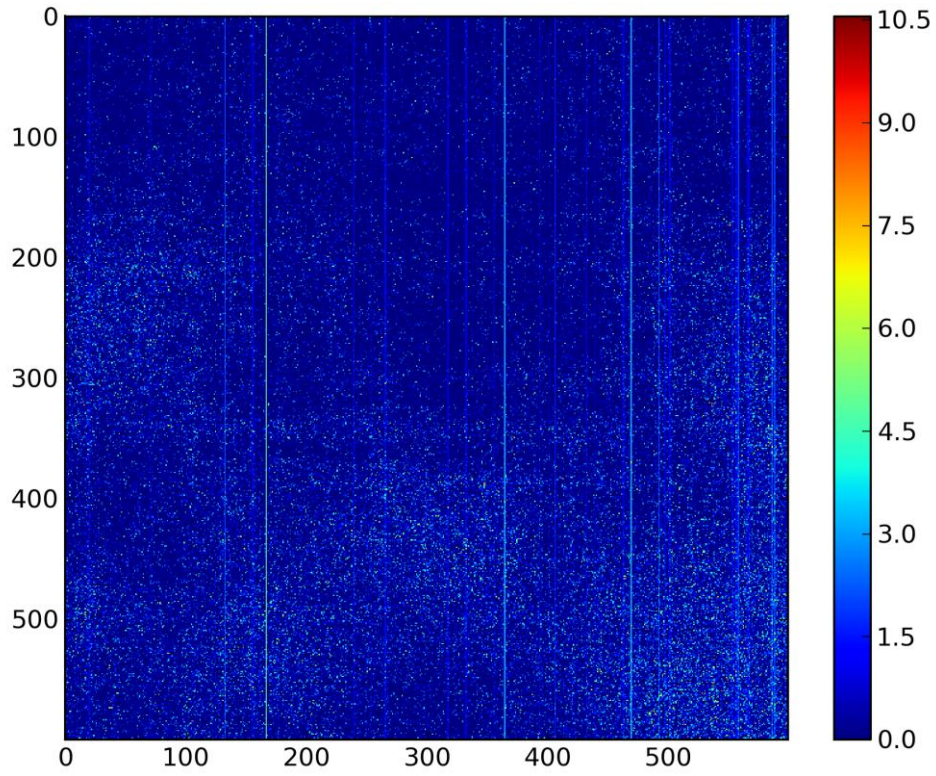


Figure 5-12: Map of the background on the sensor, obtained by time averaging a 100 frame time-series of dark images. Colour represents the counts.

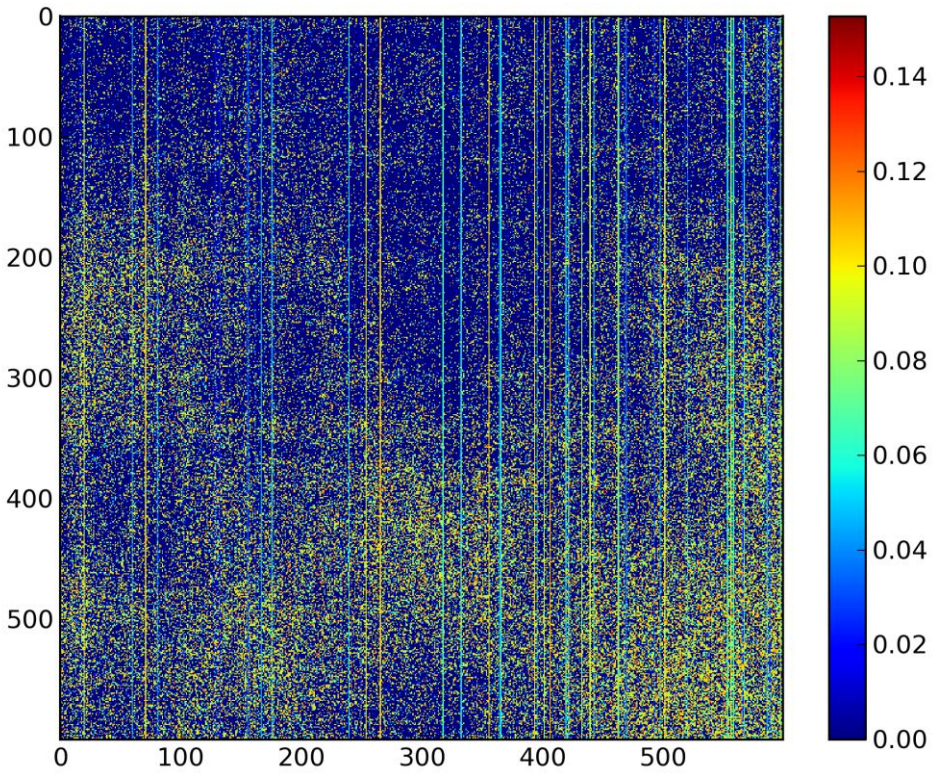


Figure 5-13: Map of the noise on the sensor obtained by applying the FDM algorithm (10 Hz window for 140 Hz centre frequency) to a 100 frame time-series of dark images. Colour represents the counts.

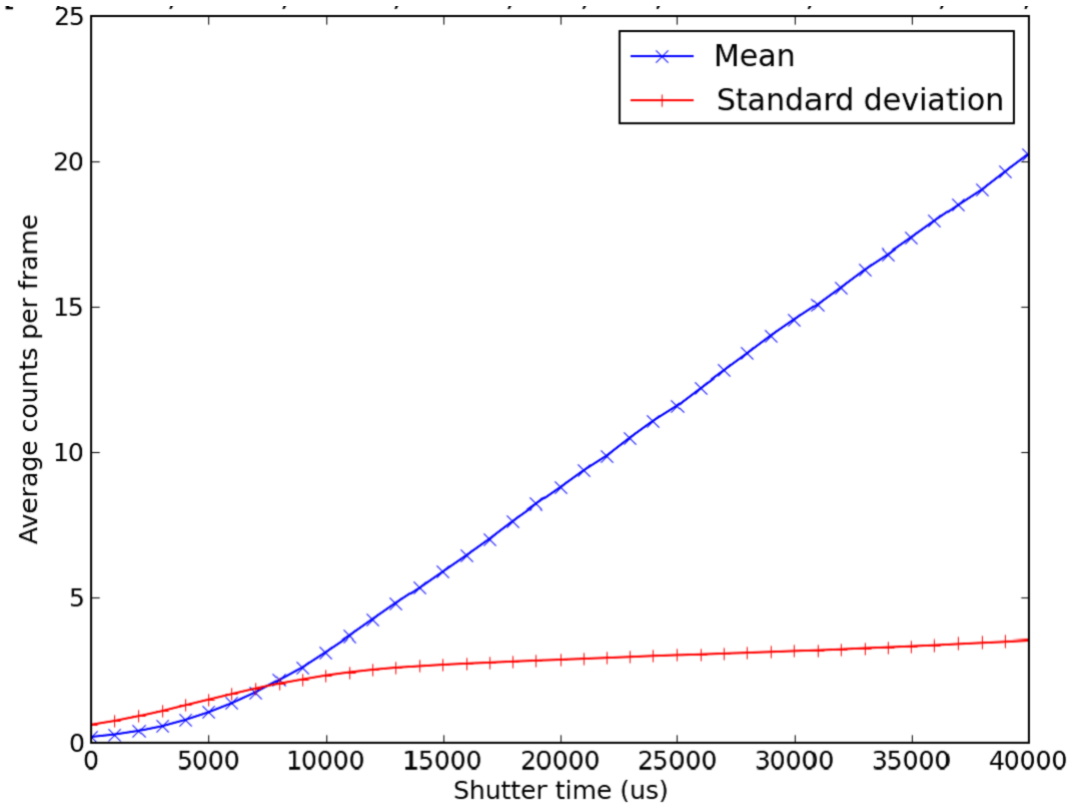


Figure 5-14: The effect of changing the shutter (integration) time of the camera (8 bit mode) on dark signal count of a typical pixel.

Image time-series were recorded with the camera lens cap on to measure directly the noise that was present in the measurements. Single frame captures at the same time as the time-series were also recorded. These data sets allowed comparison of the characteristics of the camera when taking measurements using single frames, time averaged time-series, and with application of the FDM algorithm to a time-series. The data was captured over a period of 10 minutes directly after switching on the camera and again two hours later once the camera had reached a steady temperature.

FDM was applied using the frequencies of 140 Hz and 175 Hz (the FDM shearography frequencies) for the two illuminating beams, with 10 Hz windows, and all the images captured were of dimension 600 pixels by 600 pixels at 8 bit depth. The mean of the resulting frames, either directly captured in the case of the snapped images, or the product of averaging down the time axis or through application of the FDM algorithm, was calculated to reduce the data to a single value that represented a typical pixel in the region of interest.

The noise of the data set captured just after the camera was switched on (Figure 5-15) decreased in time as the electronics warmed up. This was evident in the single frame

results, showing a strong curve downward as the time from switch-on increased. Figure 5-15 also shows the time averaged result, which followed the same downward trend as the single frame results, though the noise on the single pixel results was suppressed by the averaging of all the frames in the time-series to give a time averaged frame, equivalent to the single frame capture, of which the mean was calculated to yield the typical pixel value. It can be seen that the FDM results are unaffected by the noise during the warm up period, remaining at a near constant, low level throughout, about an order of magnitude less than that of the time averaging results, due to the algorithm removing the DC background. This is also seen in the results of the simulation described in section 5.6.1. Two hours later, when the temperature controller in the camera was stable, the results in Figure 5-16 were obtained. These show that the camera noise became stable under all three measurement regimes. Under the steady state, the noise in both single frame and time averaging was shown to be around a factor of twenty times higher than that obtained using the FDM algorithm.

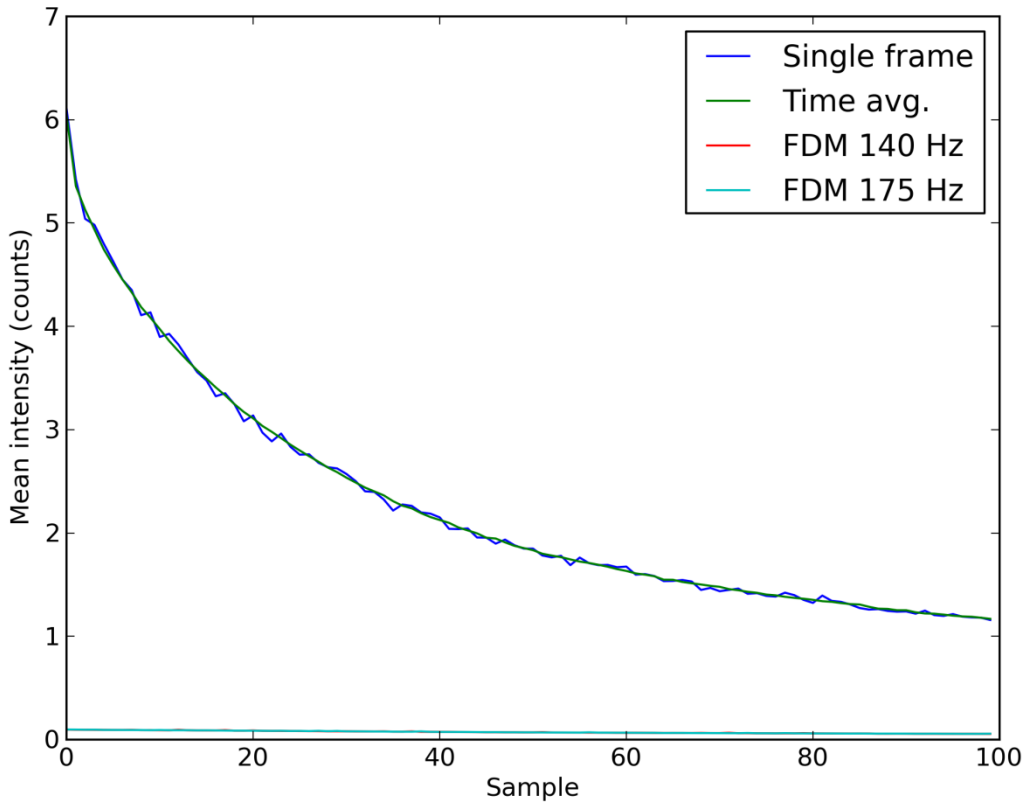


Figure 5-15: Plot of the typical pixel intensity as measured using a single frame, a time averaged time-series, and through application of the FDM algorithm at two frequencies (140 Hz and 175 Hz) just as the camera was switched on. Samples were taken over a period of ten minutes at roughly equal sample intervals of 5 s.

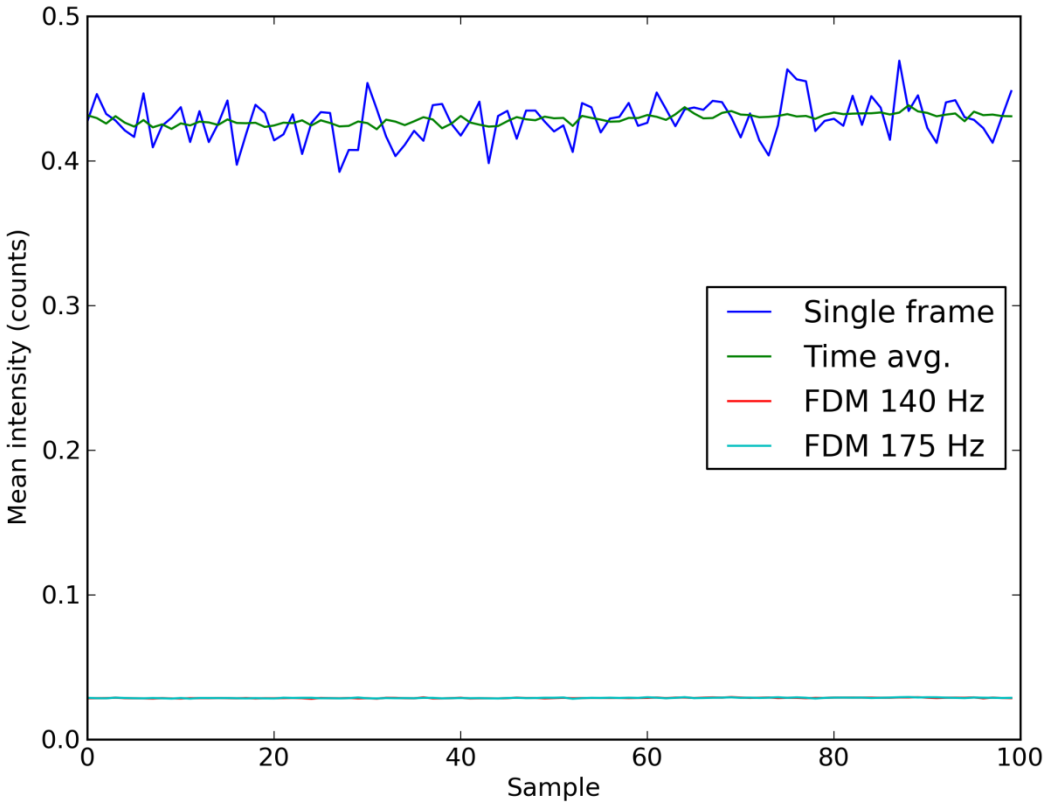


Figure 5-16: plot of the typical pixel intensity as measured using a single frame, a time averaged time-series, and through application of the FDM algorithm at two frequencies (140 Hz and 175 Hz) after the camera had been allowed to warm up for two hours and had reached a stable thermal state. Samples were taken over a period of ten minutes at roughly equal sample interview of 5 s.

5.6 Signal to noise ratios

5.6.1 Time averaging an image bank and why it's better for 1 channel

The FDM algorithm has been shown to inherently suppress the noise. In this section, simulations that aim to investigate the effects of shape of the noise in the time-series on the final calculated intensity are described. The simulations were designed to simulate a typical single pixel of an 8 bit camera under dark conditions. Simulations were conducted for increasing noise floor (the mean of the noise distribution) and for increasing spread of the noise (the standard deviation of the noise distribution) in the

time-series, and calculations were repeated 100 times so that the mean of the intensity and the spread of the repeated intensities could be calculated.

It can be seen from Figure 5-17 that increasing the level of noise in a time-series has the effect of adding a contribution to the intensity that cannot be removed by time averaging the results. However, using the FDM algorithm on the same data causes the contribution to the intensity to be constant. This demonstrates that the FDM algorithm has a clear advantage by removing the DC background.

The simulation was repeated for a fixed noise floor (20 AU) and the spread of the noise in the time-series was increased to produce the result shown in Figure 5-18. These show that increasing the standard deviation of the normally distributed noise in the time-series increases the contribution to the calculated intensity for FDM but not for time-averaging. The contribution in FDM increases linearly, though is lower than for time averaging in the range shown.

Figure 5-19 shows that increasing the standard deviation in the noise of the time-series caused the time averaged and the FDM calculated intensities to display a greater spread across the 100 repeats as the standard deviation increased. Figure 5-20 shows two time-series with the same mean level of noise, but with different standard deviations of the noise. This highlights how the noise mean may appear low but may conceal high values in the time-series. Both time averaging and FDM showed an increasing spread of results when the noise standard deviation in the time-series was increased. FDM was more stable than time averaging in this case.

While the results presented here show that time averaging can produce a lower noise than FDM, this does not tell the whole story. It is true that, for a given length of time-series in a single channel system, time averaging will give the better result. However, when the number of channels is increased, and the permitted time interval in which to capture the information from multiple channels is fixed by some experimental constraint, then the advantage of time averaging is eroded. In order to capture multiple channels in a time-averaged regime, each channel must be captured using the technique of time-division multiplexing, breaking the time-series into a number of shorter sections, one for each channel present. This is summarised in Table 5-2 where the result of experimental time-series captures processed as if for one to four channels, time-division multiplexed with time averaging to reduce noise in the result, and an FDM measurement on a time-series the length of a single time averaged channel are shown. These results represent the standard deviation of the noise on a typical pixel in the result. It can be seen that the advantage of FDM comes when more than two channels are captured. A fuller picture of the effects of noise in a system

would include a full analysis of the effect of intensity of the image signal, the noise in the time series, and the width of the windows on the final result from the phase calculation.

Table 5-2: Experimental results of noise standard deviation on a typical pixel of the Baumer HXC13 camera. Results are presented for one to four channels (128, 64, 42, and 32 frames respectively) in a time averaged 128 frame time-series, and an FDM channel from the same length of time-series with a 10 Hz window. Results are averaged over 100 repetitions.

Processing method	Standard deviation of pixel in 100 repetitions (counts)
128 frame time averaging	0.044
64 frame time averaging	0.059
128 frame FDM	0.063
42 frame time averaging	0.071
32 frame time averaging	0.081

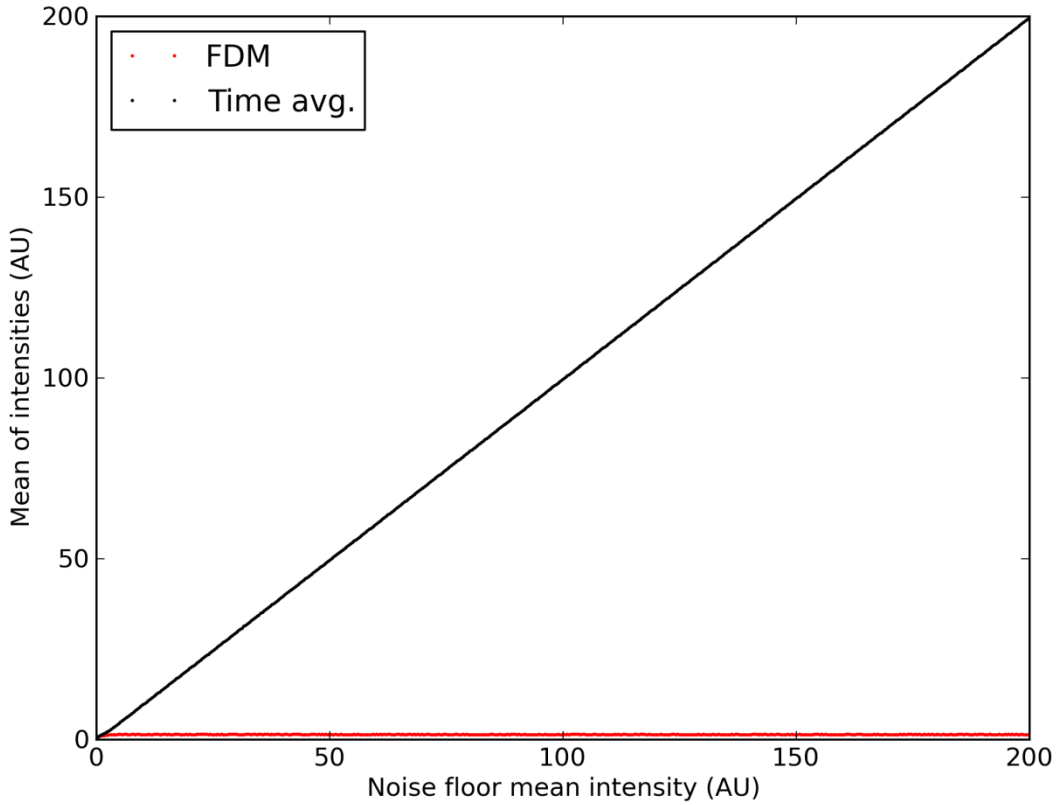


Figure 5-17: Increasing the average noise floor in the time-series and the effect on the calculated intensity from using the FDM algorithm and time averaging (Time avg.). The measurand is the mean of 100 repeats.

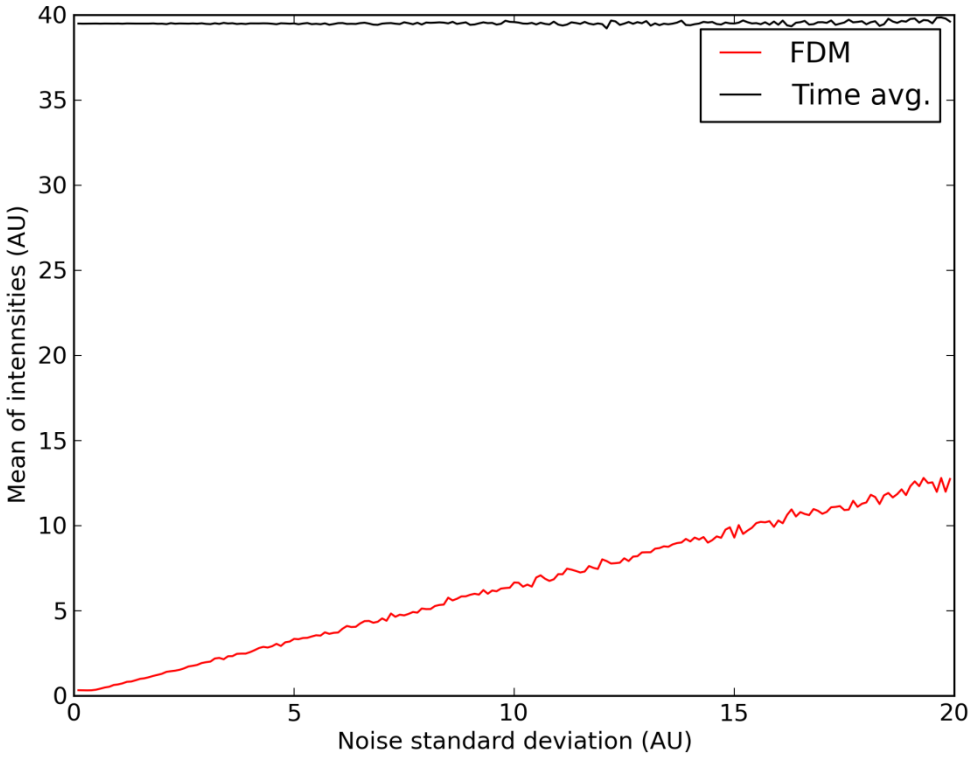


Figure 5-18: Effect on the calculated intensity of increasing the standard deviation of the noise in the time-series. Mean of 100 repeats.

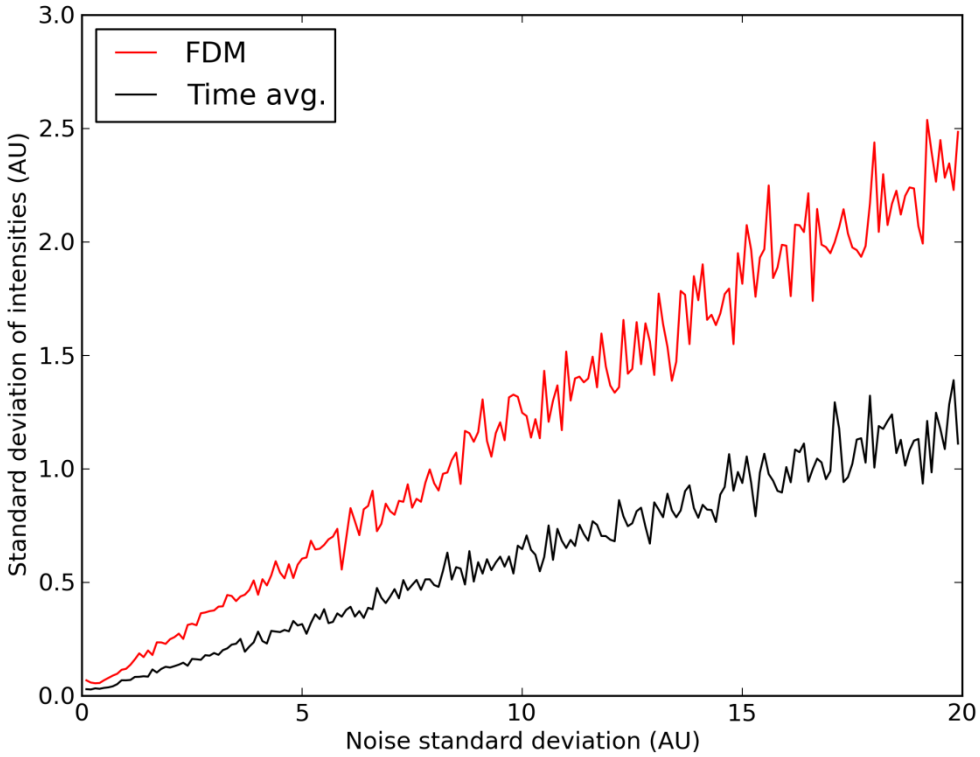


Figure 5-19: Effect on the standard deviation of the calculated intensities of 100 repeats with increasing the standard deviation of the noise in the time-series.

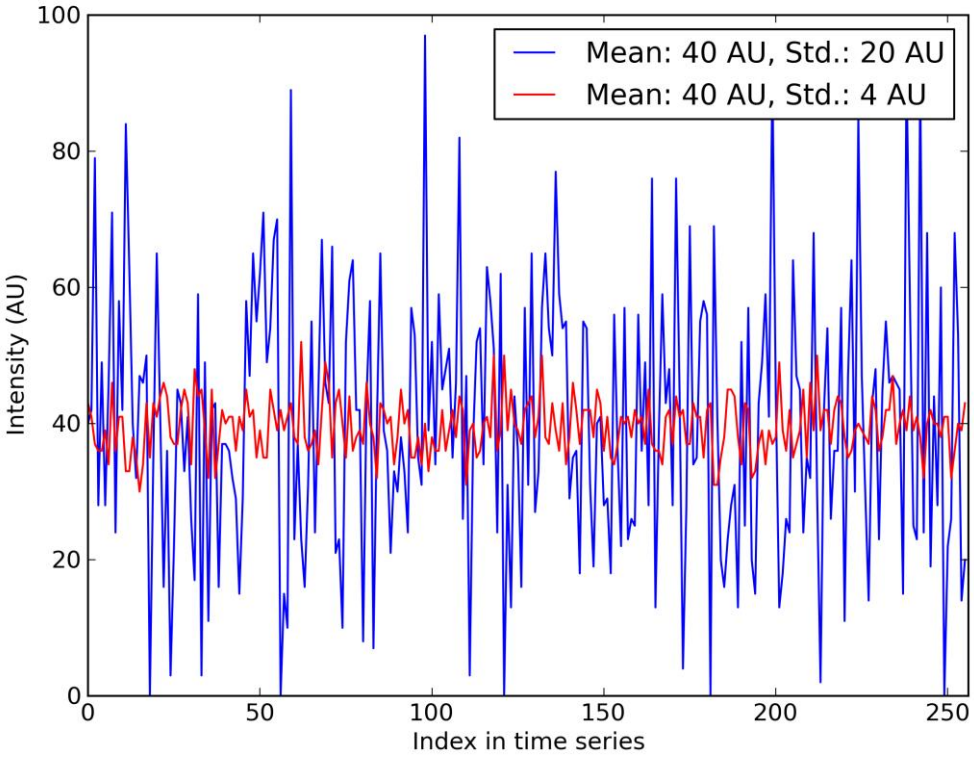


Figure 5-20: Two representative time-series both with a mean of 4 AU but with standard deviations in the noise of 4 AU and 20 AU.

5.6.2 Cross-talk

As the channels are multiplexed onto a single sensor, it would be prudent to consider the composition of a demultiplexed signal, specifically, what proportion of a signal is the desired signal and what proportion of the measurement comes from light from other channels leaking into the demultiplexing region.

To investigate this, four channels were created using the laser source and interferometer from the FDM I-PDV experiments described in chapter 4, but only the fibre delivered channel was allowed to propagate through the interferometer to the camera. The channels were created time-division multiplexed, but one at a time using the fibre channel with the beam chopper applying FDM modulations, so the four FDM channels measured in this section were temporally multiplexed rather than frequency-division multiplexed, and from these measurements, the cross-talk was measured through summation of the results. This was considered a valid approach as the signal received at the camera from a four channel system is equivalent to the summation of

the intensities of the four channels when each channel is mutually incoherent. This method also allowed camera saturation to be avoided as an issue, and so higher intensities to be captured. The channels were created by chopping the laser beam at four frequencies as might be chosen in a practical experiment, spaced approximately evenly throughout Fourier space at 5 Hz, 12 Hz, 23 Hz, and 28 Hz. The peaks resulting from each channel can be seen in Figure 5-21, overlaid with one another. The crosses show the points in the DFTs which would be included in the window of 1.5 Hz for the respective peaks. It can be seen that the peaks are positioned relative to one another so as to minimise the overlap of the fundamental peaks with harmonics and other higher order terms from the various channels. The cross-talk in this system can be read from Figure 5-22, which shows for a given channel the effect of cross-talk at each point in Fourier space. A line in this plot corresponds to the sum of three of the four channels shown in Figure 5-21, and is plotted in the colour of the omitted peak. It should be noted that, in this plot, a peak corresponds to a frequency where the cross-talk from the present channels would be at a maximum. The peaks are broadened in the figure as the spectra have been convolved with the window function to allow the cross-talk to be read for a given window centre frequency. A plot of the type shown in Figure 5-22 is, in effect, a guide to where an additional channel may be placed. This is made clearer in Figure 5-23, where the channels have been summed to one line (shown in green) and the frequency of the additional channel can be selected to be where the green line is at a minimum in order to have minimal cross-talk. The convolution of the window with the spectrum of its channel is shown in blue.

The approach of summing the convolutions of each channel present in an FDM signal, besides the channel of interest to find the low noise regions in which to place new channels can be seen from the plots shown in this section.

Cross-talk has been measured in this thesis to be 2% between channels in a two channel system and it can be seen from the plots in this section that it can reach 13% from three channels in a four channel system. This is lower than has been reported for polarisation division multiplexing of 20% [10] for two channels (multiplexed orthogonal lateral image shear directions). Therefore, in terms of signal leakage, FDM has been demonstrated to be a viable approach to channel multiplexing.

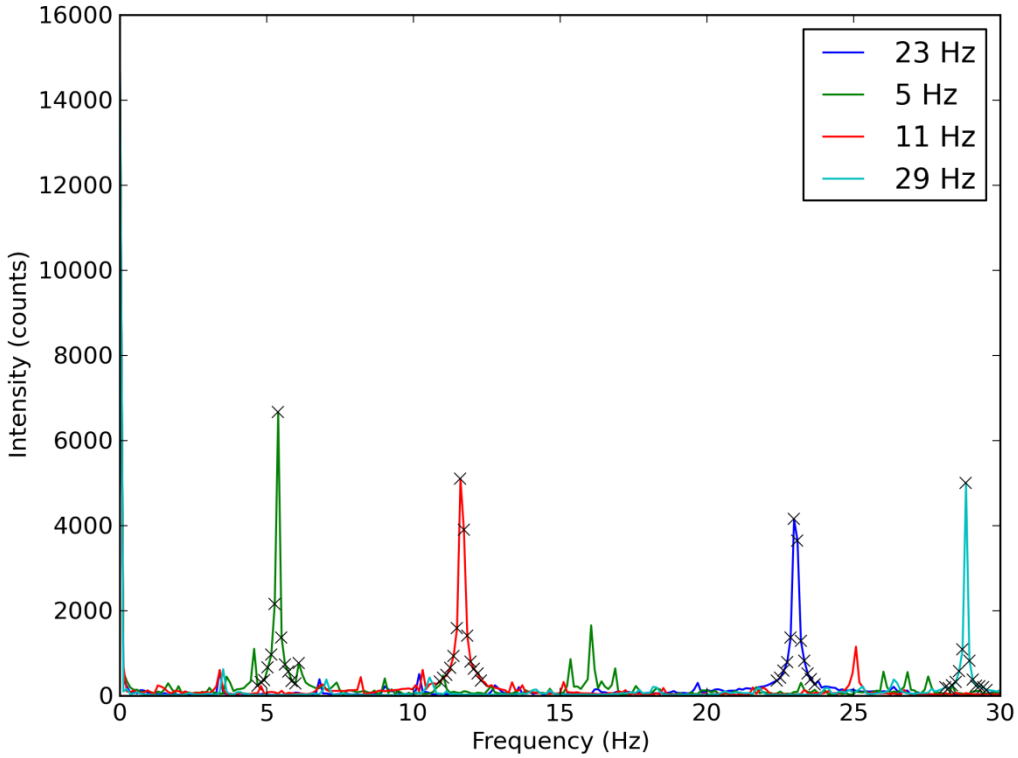


Figure 5-21: Four FDM channels in frequency space. Each line represents the signal of a single channel, and display peaks at the modulation frequency at which the beam was chopped. Crosses represent the data points that would be used by the FDM algorithm on the peaks.

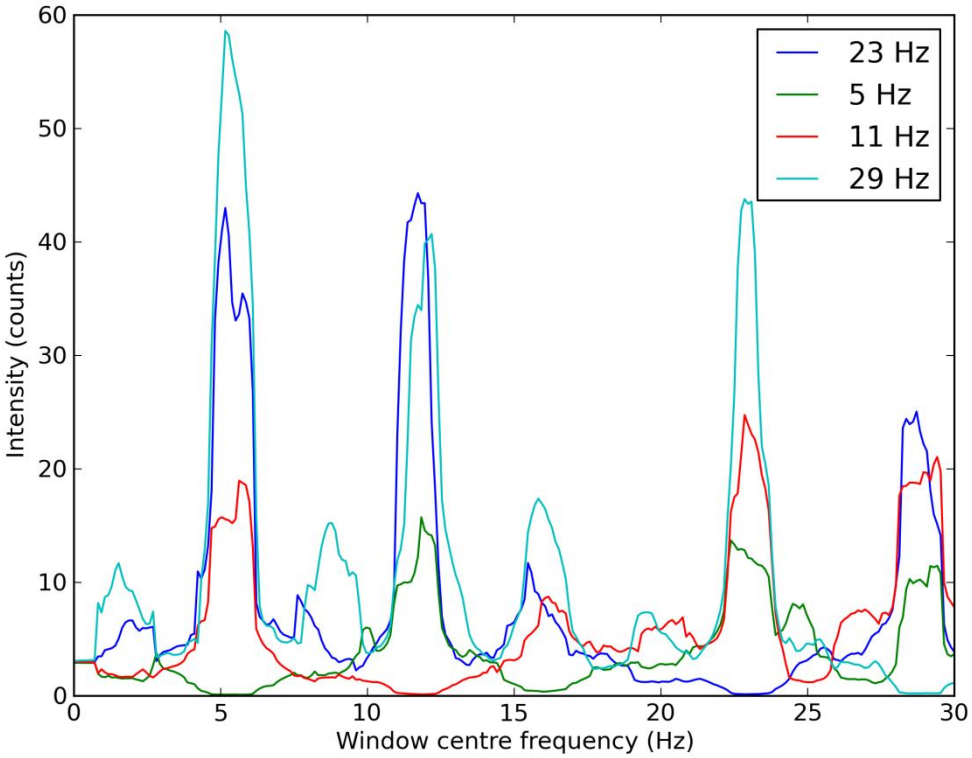


Figure 5-22: Cross-talk plot of the results shown in Figure 5-21.

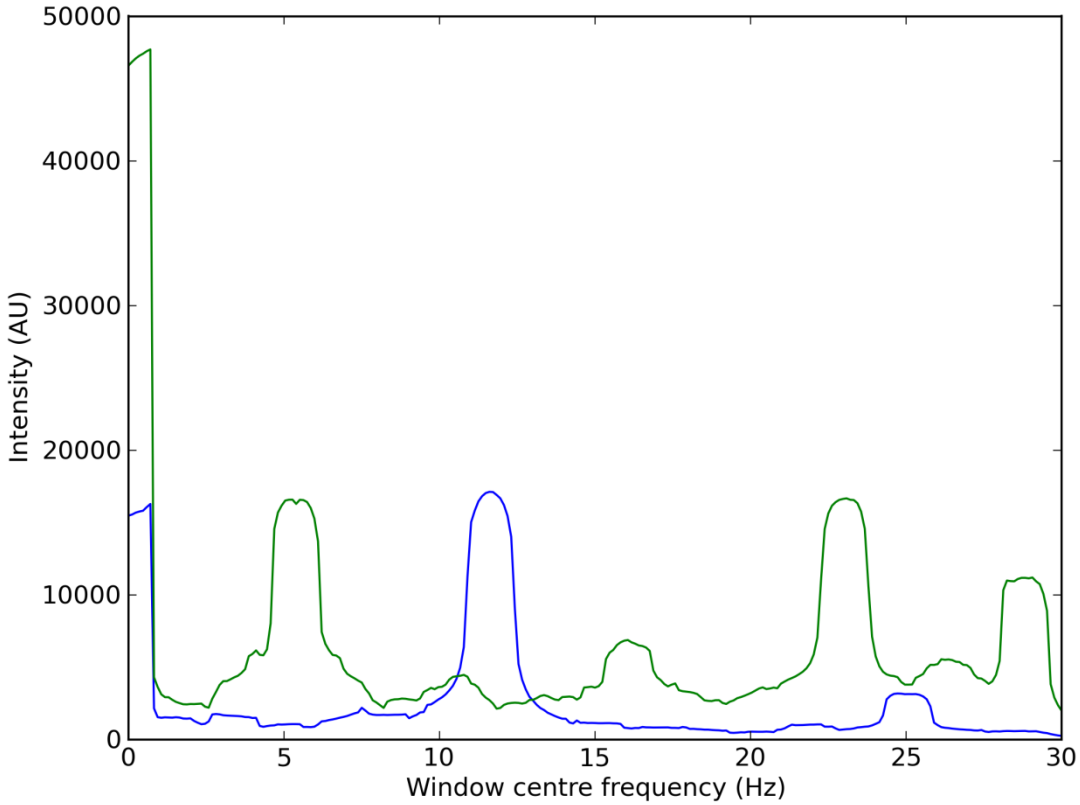


Figure 5-23: A channel at 11.5 Hz (blue) and the sum contributions of the other channels present (green) in frequency space.

5.7 Conclusion

FDM has been shown to be robust to noise sources that may be encountered during an experiment. While it was seen that, for a single channel system, FDM noise reduction was less than that of a time averaged measurement, and FDM was shown to be more repeatable than time averaging. It was shown that FDM has an excellent ability in reconstructing an image in low light conditions, though it was found to be susceptible to image phase drifts occurring during the time-series capture period, introducing a linear phase error in the reconstructed image. A method was also demonstrated to select an additional modulation frequency to fit into a multiplexed multiple channel system with minimal cross-talk. The FDM method was shown to have advantages over the time averaging and single image capture methods therefore, especially in multi-=channel systems with low illumination intensity.

5.8 Future work

While there have been demonstrations of the use of frequency-based techniques in multiplexing signals on point detectors, no prior reports of attempts to measure the typical noise values or uncertainties expected was found on such regimes, and no work at all was found on the use of these techniques with imaging sensor arrays.

The phase error discovered in image reconstruction could have serious implications on a measurement system requiring accurate measurement of full field interferometric phase when there is an associated time dependant phase error. Work would need to be undertaken to improve the algorithm's response in such circumstances.

5.9 Acknowledgement

Thanks go to Dr Charrett for the assistance in processing some of the results in this chapter, particularly those of Table 5-2 and Figure 5-6

6 Conclusions

In this thesis, a method, called frequency division multiplexing, was developed to allow the multiplexing and demultiplexing of multiple images onto a single camera sensor. This allowed the simultaneous capture of multiple channels in image based metrology techniques. The technique was applied successfully to two imaging metrology methods, that of in-plane configuration shearography and single component I-PDV.

Firstly, the development of the algorithm was described. This was based on Parseval's theorem of Fourier transforms which relates the intensity of a signal in time to the intensity of a frequency in Fourier space. It was then shown that, by modulating the intensity of light in time and capturing images of this modulation over a number of frames, the image could be reconstructed and separated from another signal. This was demonstrated successfully for the cases of the use of two separate laser sources and a single, split path laser source to illuminate the scene. Intensity modulations in time were applied through mechanical beam choppers acting upon the beams.

The FDM algorithm was applied to the case of in-plane shearography, where two laser channels were placed in a plane and imaged through an image shearing Michelson interferometer to reveal the in-plane and out-of-plane strains on a compression loaded notched plate. The use of FDM allowed the light from the two channels to be imaged simultaneously at the camera and cut down the number of steps required to make a measurement of strain, removing the need to take separate measurements for each channel which in the presented case halved the number of steps. The five-frame Schwider-Hariharan temporal phase stepping method was used to resolve the phase changes induced in the light scattered from the plate on loading and the iso-phase unwrapping algorithm was employed to remove phase wrapping. The measurements were compared to those obtained using TDM and were found to show good agreement. It was also found that noise in a reconstructed image was reduced by an order of magnitude compared to taking a single frame image. Cross-talk in the system was found to be on the order of 2% of the power in the originating channel. This work was published as [9].

FDM was applied also to another imaging metrology technique in the field of I-PDV. This measured velocity through analysing Doppler shifts in light scattered from the moving body via imaging through an MZI onto the camera. Doppler shift was measured as a phase change in the light, obtained using the carrier fringe phase measurement method. I-PDV measurements were made of the velocity on the surface of a spinning disc with a horizontal velocity component of ~40 ms^{-1}, and of the velocity

of a circular free jet of air at ~80 ms^{-1} in a background flow of ~20ms^{-1}. The system contained two channels, one with light fibre-delivered to the interferometer directly from the laser source as a reference, and the other coming from the light transmitted to and then scattered by the flow, and these were multiplexed together with FDM using beam choppers. This allowed single component measurements of the velocities. It was found that the positioning of the windows around the carrier fringe peaks in the DFT could adversely affect the result. Information could be lost by over-filtering when using small windows around the peaks, though increasing the area of the windows from a tight 20 x 20 pixel crop around the peak to a larger area of 450 x 50 pixel allowed more features in the velocity to be observed, it also allowed more noise. When the results between FDM and TDM measurements of the disc were compared, it was found that the FDM measurements were closer to the theoretical velocity than those of TDM due to the temporal drifting of phase appearing as a Doppler shift and influencing the velocity calculation. In the case of the jet, the velocity was measured using I-PDV and compared to a model using measurements of the jet velocity obtained using a Pitot tube. The I-PDV measurements were found to follow the modelled form. It was noted that fringe phase drifts could introduce noise into the velocities when they got large enough to strongly reduce fringe visibility. The FDM technique was found to be good at suppressing the background signal and noise in measurements. It was also found to be effective at reconstructing low intensity images. The advantage of the FDM technique was found to be in application to multiple channel systems with more than three channels. In this way, multiple measurements may be taken from each channel, simultaneously rather than sequentially, and at full camera resolution. With a spatial phase encoding technique applied, measurements may be taken in the same time period for a less noisy result from multiple channels.

References

[1] A. Fischer, L. Buttner and J. Czarske, “Simultaneous measurements of multiple flow velocity components using frequency modulated lasers and single absorption cell,” *Optics Communications,* vol. 284, pp. 3060-3064, 2011.

[2] M. Takeda, H. Ina and S. Kobayashi, “Fourier-transform method of fringe-pattern analysis for computer based topography and interferometry,” *J. Opt. Soc. Am.,* vol. 72, no. 1, pp. 156-160, 1982.

[3] J. M. Blackledge, Quantitative Coherent Imaging, London: Academic Press, 1989.

[4] K. Creath, “Temporal phase measurement methods,” in *Interferogram analysis: Digital fringe measurement methods*, D. W. Robinson and G. T. Reid, Eds., Bristol, UK, IOP Publishing, 1993, pp. 94-140.

[5] Python Software Foundation, August 2011. [Online]. Available: http://www.python.org.

[6] T. O. H. Charrett, “Python Toolkit,” April 2011. [Online]. Available: http://pythontoolkit.sourceforge.net/.

[7] E. Jones, T. Oliphant, P. Peterson et al., August 2011. [Online]. Available: http://www.scipy.org.

[8] M. Frigo and S. G. Johnson, “Fastest Fourier Transform in the West,” May 2011. [Online]. Available: http://www.fftw.org.

[9] I. A. Bledowski, T. O. H. Charrett, D. Francis, S. W. James and R. P. Tatam, “Frequency division multiplexing for multi-component shearography,” *Appl. Opt.,* vol. 53, no. 3, pp. 350-358, 2013.

[10] R. M. Groves, S. W. James and R. P. Tatam, “Polarization-multiplexed and phase-stepped fibre optic shearography using laser wavelength modulation,” *Meas. Sci. Technol.,* vol. 11, no. 9, pp. 1389-1395, 2000.

[11] D. Francis, R. P. Tatam and R. M. Groves, “Shearography technology and applications: A review,” *Meas. Sci. Technol.,* vol. 21, no. 10, 102001, 2010.

[12] M. Kalms and W. Osten, “Mobile shearography system for the inspection of aircraft and automotive components,” *Opt. Eng.,* vol. 42, no. 5, pp. 1188-1196, 2003.

[13] U. Paul Kumar, M. P. Kothiyal and N. Krishna Mohan, “Microscopic TV shearography for characterization of microsystems,” *Opt. Lett.,* vol. 34, no. 10, pp. 1612-1614, 2009.

[14] W. Steinchen and L. Yang, Digital shearography, Washington: SPIE, 2003.

[15] Y. H. Huang, L. Liu, Y. S. Chen, C. L. Li and Y. Y. Hung, “Unified approach for rough phase measurement without phase unwrapping by changing the sensitivity factor,” *J. Mod. Opt.,* vol. 56, no. 9, pp. 1070-1077, 2009.

[16] J. W. Goodman, Speckle phenomena in optics: theory and applications, Englewood: Roberts and Company Publishers, 2007.

[17] J. H. Bruning, D. R. Herriott, J. E. Gallagher, D. P. Rosenfeld, A. D. White and D. J. Brangaccio, "Digital wavefront measuring interferometer for testing optical surfaces and lenses," *Appl. Opt.,* vol. 13, no. 11, pp. 2693-2703, 1974.

[18] S. Nakadate and H. Saito, "Fringe scanning speckle-patter interferometry," *Appl. Opt.,* vol. 24, no. 14, pp. 2172-2180, 1985.

[19] M. R. Viotti, A. G. Albertazzi Jr. and W. Kapp, "Experimental comparison between a portable DSPI device with diffractive optical element and a hole drilling strain gauge combined system," *Opt. Laser Eng.,* vol. 46, no. 11, pp. 835-841, 2008.

[20] W. D. Pilkey, Formulas for stress, strain and structural matrices, 2nd ed., Hoboken, NJ: John Wiley & Sons, 2005.

[21] D. Francis, Surface strain measurement using pulsed laser shearography with fibre-optic imaging bundles (PhD thesis), Cranfield, UK: Cranfield University, 2008.

[22] J. A. Leendertz and J. H. Butters, "An image-shearing speckle-pattern interferometer for measuring bending moments," *J. Phys. E.: Sci. Instrum.,* vol. 6, pp. 1107-1110, 1973.

[23] Y. Y. Hung, J. L. Turner, M. Tafralian, J. D. Hovanesian and C. E. Taylor, "Optical method for measuring contour slopes of an object," *Appl. Opt.,* vol. 17, no. 1, pp. 128-131, 1978.

[24] Y. Y. Hung and R. M. Grant, "Shearography: a new optical method for nondestructive evaluation of tires," *Rubber Chemistry and Technology,* vol. 54, no. 5, pp. 1042-1050, 1981.

[25] I. Balboa, H. D. Ford and R. P. Tatam, "Low-coherence optical fibre speckle interferometry," *Meas. Sci. Technol.,* vol. 17, no. 4, pp. 605-616, 2006.

[26] W. Steinchen, L. X. Yang and M. Schuth, "TV-shearography for measuring for measuring 3D-strains," *Strain,* vol. 32, pp. 49-57, 1996.

[27] S. W. James and R. P. Tatam, "3D shearography for surface strain analysis," *Proc. SPIE,* vol. 3783, pp. 247-256, 1999.

[28] T. O. H. Charrett, D. Francis and R. P. Tatam, "Quantitative shearography: error reduction by using more than three measurement channels," *Appl. Opt.,* vol. 50, no. 2, pp. 134-146, 2011.

[29] S. W. James and R. P. Tatam, "Time-division-multiplexed 3D shearography," *Proc. SPIE,* vol. 3744, pp. 394-403, 1999.

[30] D. T. Goto and R. M. Groves, "Error analysis of 3D shearography using finite-element modelling," *Proc. SPIE,* vol. 7718, no. 771816-1, 2010.

[31] H. A. Aebischer and S. Waldner, "Strain distributions made visible with image-shearing speckle pattern interferometry," *Opt. Las. Eng.,* vol. 26, pp. 407-420,

1997.

[32] R. M. Groves, S. W. James and R. P. Tatam, “Strain measurement in curved industrial components using multicomponent shearography,” *Proc. SPIE,* vol. 4398, pp. 216-224, 2001.

[33] A. Fischer, L. Buttner, J. Czarske, M. Eggart, G. Grosche and H. Muller, “Investigation of time-resolved single detector Doppler global velocimetry using sinusoidal laser frequency modulation,” *Meas. Sci. Technol.,* vol. 18, no. 8, pp. 2529-2545, 2007.

[34] D. Francis, S. W. James and R. P. Tatam, “Surface strain measurement of rotating objects using pulsed laser shearography with coherent fibre-optic imaging bundles,” *Meas. Sci. Technol.,* vol. 19, no. 10, 105301, 2008.

[35] S. M. Soloff, R. J. Adrian and Z.-C. Liu, “Distortion compensation for stereoscopic particle image velocimetry,” *Meas. Sci. Technol.,* vol. 8, p. 1441–1454, 1997.

[36] T. Yatagai, “Intensity based analysis methods,” in *Interferogram analysis: Digital fringe pattern measurement techniques*, Bristol, UK, IOP Publishing, 1993, pp. 72-93.

[37] P. Carré, “Installation et utilisation du comparateur photoélectrique et interférentiel du Bureau International des Poids et Mesures,” *Metrologia,* vol. 2, no. 1, pp. 13-23, 1966.

[38] J. C. Wyant, “Interferometric and optical metrology: Basic principles and new systems,” *Laser Focus with Fiberoptic Technology,* vol. 18, no. 5, pp. 65-71, 1882.

[39] J. Schmit and K. Creath, “Window function influence on phase error in phase-shifting algorithms,” *Appl. Opt.,* vol. 35, no. 28, pp. 5624-5649, 1996.

[40] J. Schwider, R. Burow, K.-E. Elssner, J. Grzanna, R. Spolaczyk and K. Merkel, “Digital wave-front measuring interferometry: some systematic error sources,” *Appl. Opt.,* vol. 22, no. 21, pp. 3421-3432, 1983.

[41] P. Hariharan, B. F. Oreb and T. Eiju, “Digital phase-shifting interferometry: A simple error compensating phase calculation algorithm,” *Appl. Opt.,* vol. 26, no. 13, pp. 2504-2506, 1987.

[42] J. Angel and P. L. Wizinowich, “A method for phase shifting interferometry in the presence of vibration: a new algorithm and system,” *ESO Proceedings,* vol. 30, pp. 561-567, 1988.

[43] P. L. Wizinowich, “A method for phase shifting interferometry in the presence of vibration: a new algorithm and system,” *Appl. Opt.,* vol. 29, no. 22, pp. 3271-3279, 1990.

[44] E. Vikhagen, “Nondestructive testing by use of TV holography and deformation phase gradient,” *Appl. Opt.,* vol. 29, no. 1, pp. 137-144, 1990.

[45] M. A. Herráez, M. A. Gdeisat, D. R. Burton and M. J. Lalor, “Robust, fast, and effective two-dimensional automatic phase unwrapping algorithm based on image

decomposition," *Appl. Opt.,* vol. 41, pp. 7445-7455, 2002.

[46] D. C. Ghiglia and M. D. Pritt, Two-dimensional phase unwrapping: Theory, applications and software, New York: John Wiley & Sons, 1998.

[47] D. W. Robinson, "Phase unwrapping methods," in *Interferogram analysis: Digital fringe pattern measurement techniques*, Bristol, IOP Press, 1993, pp. 194-229.

[48] J. Schorner, A. Ettemeyer, U. Neupert, H. Rottenkolber, C. Winter and P. Obermeier, "New approaches in interpreting holographic images," *Opt. Las. Eng.,* vol. 14, pp. 283-291, 1991.

[49] O. Y. Kwon, "Advanced wavefront sensing at Lockheed," *Proc. SPIE,* vol. 816, pp. 196-211, 1987.

[50] J. Angel and P. L. Wizinowich, "Phase shifting interferometry in the presence of vibration: a new algorithm and system," *ESO Proceedings,* vol. 30, pp. 561-567, 1988.

[51] D. Derauw, "Phase unwrapping using coherence measurements," *Proc. SPIE,* vol. 2584, pp. 319-324, 1995.

[52] T. J. Flynn, "Consistent 2-D phase unwrapping guided by a quality map," in *Proceedings of the 1996 International Geoscience and Remote Sensing Symposium*, Lincoln, NE, 1996.

[53] K. Crennell, "Introductory digital image processing," in *Interferogram analysis: Digital fringe measurement methods*, Bristol, IOP, 1993, pp. 1-22.

[54] A. Aebisher and S. Waldner, "A simple and effective method for filtering speckle-interferometric phase patterns," *Opt. Comm.,* vol. 162, pp. 205-210, 1999.

[55] R. M. Groves, E. Chehura, W. Li, S. E. Staines, S. W. James and R. P. Tatam, "Surface strain measurement: A comparison of speckle shearing interferometry and optical fibre Bragg gratings with resistance foil strain gauges," *Meas. Sci. Technol.,* vol. 18, no. 5, pp. 1175-1184, 2007.

[56] F. C. I. Catalan, R. D. Santos and P. F. Almoro, "Complete deformation analysis of transparent samples using digital shearography and holography," *Proc. SPIE,* vol. 7155, no. 715536, 2008.

[57] F. Seffrin, F. Fuest, D. Geyer and A. Dreizler, "Flow field studies of a new series of turbulent premixed stratified flames," *Combustion and Flame,* vol. 157, no. 2, pp. 384-396, 2010.

[58] M. Samimy and M. P. Wernet, "Review of planar multiple-component velocimetry in high-speed flows," *AIAA Journal,* vol. 38, no. 4, pp. 553-574, 2000.

[59] G. S. Elliott and T. J. Beutner, "Molecular filter based planar Doppler velocimetry," *Progress in Aerospace Sciences,* vol. 35, pp. 799-845, 1999.

[60] F. Seiler, A. George, J. Srulijes and M. Havermann, "Progress in Doppler picture velocimetry (DPV)," *Exp. Fluids,* vol. 44, pp. 389-395, 2008.

[61] Z.-H. Lu, T. O. H. Charrett, H. D. Ford and R. P. Tatam, “Mach-Zehnder interferometric filter based planar Doppler velocimetry (MZI-PDV),” *J. Opt. A.,* vol. 9, no. 11, pp. 1002-1013, 2007.

[62] J. N. Forkey, Development and demonstration of filtered Rayleigh scattering - a laser based flow diagnostic for planar measurement of velocity, temperature and pressure (PhD thesis), Princeton, NJ: Princeton University, 1996.

[63] A. Pichler, A. George, F. Seiler, J. Srulijes and B. Sauerwein, “Doppler picture velocimetry applied to hypersonics: automated DPV fringe pattern analysis using the FFT method,” *Shock Waves,* vol. 19, pp. 413-421, 2009.

[64] M. Kujawinska, “Spatial phase measurement methods,” in *Interferogram analysis: Digital fringe pattern measurement techniques*, London, Institute of Physics Publishing, 1993, pp. 141-144.

[65] Z.-H. Lu, T. O. H. Charrett and R. P. Tatam, “Three-component planar velocimetry measurements using Mach-Zehnder interferometric filter-based planar Doppler velocimetry (MZI-PDV),” *Meas. Sci. Technol.,* vol. 20, no. 3, 034019, 2009.

[66] A. Landolt and T. Roesgen, “Global Doppler frequency shift detection with near-resonant interferometry,” *Exp. Fluids,* vol. 47, no. 4-5, pp. 733-743, 2009.

[67] M. S. Reinath, “Doppler global velocimeter development for large wind tunnels,” *Meas. Sci. Technol.,* vol. 12, no. 4, pp. 432-441, 2001.

[68] T. O. H. Charrett and R. P. Tatam, “Single camera three component planar velocity measurements using two frequency Planar Doppler Velocimetry (2ν-PDV),” *Meas. Sci. Technol.,* vol. 17, no. 5, pp. 1194-1206, 2006.

[69] A. Fischer, L. Buttner, J. Czarske, M. Eggert and H. Muller, “Measurements of velocity spectra using time-resolving Doppler global velocimetry with laser frequency modulation and a detector array,” *Experimental Fluids,* vol. 47, pp. 599-611, 2009.

[70] G. Frankowski, I. Stobbe, W. Tischer and F. Schillke, “Investigation of surface shapes using a carrier frequeny based analysing system,” *Proc. SPIE,* vol. 1121, no. Interferometry '89, pp. 89 - 100, 1989.

[71] J. A. Quiroga, J. A. Gomez-Pedrero and A. Garcia-Botella, “Algorithm for fringe pattern normalization,” *Opt. Comm.,* vol. 197, no. 1-3, pp. 43-51, 2001.

[72] C. Willert, “Stereoscopic digital particle image velocimetry for application in wind tunnel flows,” *Meas. Sci. Technol.,* vol. 8, no. 12, pp. 1465-1479, 1997.

[73] N. Rajaratnam, Developments in water science 5: Turbulent jets, Amsterdam: Elsevier, 1976.

Appendix A: Custom C library source code

This appendix contains the source code for the library (fdm.dll) used to process time-series data when applying the FDM algorithm. This library was written to utilise the speed of the C language and control the memory usage during data processing. The algorithm was accessed through the Python module

```
/*
C library to process and analyse shearography data in lock-in
approach, returning speckle patterns.
Functions provide speed increase over Python implementation.
*/

/*
On windows _declspec(dllexport) makes the function visable without a
.def file
for linux it is not needed, so use these lines to allow easy
compilation on both
*/
#ifdef WIN32
#define EXPORTED __declspec(dllexport)
#else
#define EXPORTED
#endif

#include <math.h>
#include <float.h>
#include <stdlib.h>
#include <string.h>
#include <fftw3.h>


/*DEMUX SECTION START*/

/*Rayleigh analysis on captured data (int8) and image extraction
function
    in                  int8 array      Captured image stack
    inz, iny, inx       int (32bit)     Size dimensions of image stack
    no_masks            int (32 bit)    Number of frequencies to demux
    freq_masks          pointer         Some pointer to an array of
frequency masks (doubles) for each mux frequency, each of length inz.
    images              pointer         Pointers to the (double)
arrays for each demuxed image to be output to, in same order as freqs.
*/

/*
demux_lite
```

```
Does the minimum amount of work to obtain demuxed images from an FDM
image bank. Results are scaled non-linearly in intensity as
square-root is NOT taken
This will be the fastest method.
*/
EXPORTED int demux_lite(unsigned char *in, int inz, int iny, int inx,
int no_masks, double *freq_masks, double *images)
{
    fftw_complex *fft_array;
    fftw_plan plan;
    double rayleigh, *time_series;
    int x,y,z,mask,Nt;

    Nt = (inz/2) + 1; //length of the r2c transform

    time_series = (double*) fftw_malloc( sizeof(double)*inz);
//working array in fftw_complex length of timeseries z
    fft_array = (fftw_complex*) fftw_malloc( sizeof(fftw_complex) *
Nt);             //out array in fftw_complex length of timeseries z

    plan = fftw_plan_dft_r2c_1d(inz, time_series, fft_array,
FFTW_EXHAUSTIVE);        //real to complex dft

    for (y=0; y<iny; y++)
    {
        for (x=0; x<inx; x++)
        {//Do fft on pixel timeseries
            for (z=0; z<inz; z++)
            {//Build time array from in
                time_series[z] =
(double)(*(in+(iny*inx*z)+(inx*y)+x));
            }
            fftw_execute_dft_r2c(plan, time_series, fft_array);

            //Construct image for each demux frequency == a mask
            for (mask=0; mask<no_masks; mask++)
            {
                rayleigh = 0;
                for (z=1; z<Nt; z++)
                {// only need to loop over the positive frequencye as
fft symmetric.
                    if (*(freq_masks+(mask*Nt)+z) > 0) //only include
values where mask is 1
                    {
                        rayleigh = rayleigh + (
*(freq_masks+(mask*Nt)+z)*(fft_array[z][0]*fft_array[z][0] +
fft_array[z][1]*fft_array[z][1]) );
                    }
                }
```

```
                //store sum into the images array, x2 to take into
account the negative frequencies that arent calculated in the fft and
arent summed above.
                *(images+(mask*iny*inx)+(inx*y)+x) = 2 * rayleigh;
            }
        }
    }
    //free resources
    fftw_free(time_series);
    fftw_free(fft_array);
    fftw_destroy_plan(plan);

    return 1;
}

/*
demux
Obtains demuxed images from an FDM image bank. Results are scaled
linearly in intensity as square-root is taken.
This will therefore be the slower than demux_lite.
*/
EXPORTED int demux(unsigned char *in, int inz, int iny, int inx, int
no_masks, double *freq_masks, double *images)
{
    fftw_complex *fft_array;
    fftw_plan plan;
    double rayleigh, *time_series;
    int x,y,z,mask,Nt;

    Nt = (inz/2) + 1; //length of the r2c transform

    time_series = (double*) fftw_malloc( sizeof(double)*inz);
//working array in fftw_complex length of timeseries z
    fft_array = (fftw_complex*) fftw_malloc( sizeof(fftw_complex) *
Nt);             //out array in fftw_complex length of timeseries z

    plan = fftw_plan_dft_r2c_1d(inz, time_series, fft_array,
FFTW_EXHAUSTIVE);        //real to complex dft

    for (y=0; y<iny; y++)
    {
        for (x=0; x<inx; x++)
        {//Do fft on pixel timeseries
            for (z=0; z<inz; z++)
            {//Build time array from in
                time_series[z] =
(double)(*(in+(iny*inx*z)+(inx*y)+x));
            }
            fftw_execute_dft_r2c(plan, time_series, fft_array);
```

```
            //Construct image for each demux frequency == a mask
            for (mask=0; mask<no_masks; mask++)
            {
                rayleigh = 0;
                for (z=1; z<Nt; z++) //IMPORTANT: this will NEVER
include DC term - means can just x2 the rayleigh value later via
symmetry.
                {// only need to loop over the positive frequencye as
fft symmetric.
                    if (*(freq_masks+(mask*Nt)+z) > 0) //only include
values where mask is 1
                    {
                        rayleigh = rayleigh + (
*(freq_masks+(mask*Nt)+z)*(fft_array[z][0]*fft_array[z][0] +
fft_array[z][1]*fft_array[z][1]) );
                    }
                }
                //store Rayleigh sum into the images array, times two
and scaled (essentially the RMS value of the modulation signal in the
window).
                *(images+(mask*iny*inx)+(inx*y)+x) = sqrt((2 *
rayleigh)/(inz*inz));
            }
        }
    }
    //free resources
    fftw_free(time_series);
    fftw_free(fft_array);
    fftw_destroy_plan(plan);

    return 1;
}

/*
demux_complex
This version uses the complex to complex fft and scales the result by
square-root to produce linear intensity images.
Only present for interest, not expected to use this on FDM data which
is wholly real.
*/
EXPORTED int demux_complex(unsigned char *in, int inz, int iny, int
inx, int no_masks, double *freq_masks, double *images)
{
    fftw_complex *time_series, *fft_array;
    fftw_plan plan;
    double rayleigh;
    int x,y,z,mask;
```

```
    time_series = (fftw_complex*) fftw_malloc(
sizeof(fftw_complex)*inz);         //working array in fftw_complex
length of timeseries z
    fft_array = (fftw_complex*) fftw_malloc(
sizeof(fftw_complex)*inz);             //out array in fftw_complex
length of timeseries z

    plan = fftw_plan_dft_1d(inz, time_series, fft_array, FFTW_FORWARD,
FFTW_EXHAUSTIVE); //complex to complex dft

    for (y=0; y<iny; y++)
    {
        for (x=0; x<inx; x++)
        {//Do fft on pixel timeseries
            for (z=0; z<inz; z++)
            {//Build time series array
                time_series[z][0] = *(in+(iny*inx*z)+(inx*y)+x);
                time_series[z][1] = 0;
            }
            fftw_execute_dft(plan, time_series, fft_array);

            //Construct image for each demux frequency == a mask
            for (mask=0; mask<no_masks; mask++)
            {
                rayleigh = 0;
                for (z=0; z<inz; z++)
                {// |F|^2 == (a+ib)(a-ib) == a^2 + b^2
                    if (*(freq_masks+(mask*inz)+z) > 0)
                    {
                        rayleigh = rayleigh +
(*(freq_masks+(inz*mask)+z)*(fft_array[z][0]*fft_array[z][0] +
fft_array[z][1]*fft_array[z][1]));
                    }
                }
                //store sum into the images array, scaled by length of
transform and square-rooted.
                *(images+(mask*iny*inx)+(inx*y)+x) =
sqrt(rayleigh/(inz*inz));
            }
        }
    }
    //free resources
    fftw_free(time_series);
    fftw_free(fft_array);
    fftw_destroy_plan(plan);

    return 1;
}
```

```
/*END OF DEMUX SECTION*/


/*
Compute the FFTW real to complex transform on a image bank, in(t,y,x)
and store in image bank, out(f,y,x)
size of in:    inz,iny,inx  of type=int32
size of out:   (inz/2)+1,iny,inx of type=complex128 (standard numpy
complex type)
*/
EXPORTED int fft(unsigned char *in, int inz, int iny, int inx,
fftw_complex *out)
{
    //timeseries dimensions
    int Nt = ((inz/2)+1); //length of fft (positive freqs only)

    //for the fft
    double *time_series;        //time series array
    fftw_complex *fft_array;    //fft array
    fftw_plan plan;             //fft plane

    time_series = (double*) fftw_malloc( sizeof(double)*inz);
//working array in fftw_complex length of timeseries z
    fft_array = (fftw_complex*) fftw_malloc( sizeof(fftw_complex)*Nt);
//out array in fftw_complex length Nt
    plan = fftw_plan_dft_r2c_1d(inz, time_series, fft_array,
FFTW_EXHAUSTIVE);      //real to complex dft

    //Loop over pixels
    int x,y,z;
    for (y=0; y<iny; y++)
    {
        for (x=0; x<inx; x++)
        {

            //Do fft on pixel timeseries
            for (z=0; z<inz; z++)
            {
                //Build time array from in
                time_series[z] =
(double)(*(in+(iny*inx*z)+(inx*y)+x));
            }
            fftw_execute_dft_r2c(plan, time_series, fft_array);

            //copy to output array
            for (z=0; z<Nt; z++)
            {
                //store into out array
                out[ (iny*inx*z)+(inx*y)+x ][0] = fft_array[z][0];
                out[ (iny*inx*z)+(inx*y)+x ][1] = fft_array[z][1];
```

```
            }
        }
    }
return 1;
}

/*
Compute the power spectrums using the FFTW real to complex transform
on a image bank, in(t,y,x) and store in image bank, out(f,y,x)
size of in:    inz,iny,inx  of type=int32
size of out:   (inz/2)+1,iny,inx of type=double (the standard numpy
array dtype)
*/
EXPORTED int power_spectrum(unsigned char *in, int inz, int iny, int
inx, double *out)
{
    //timeseries dimensions
    int Nt = ((inz/2)+1); //length of fft (positive freqs only)

    //for the fft
    double *time_series;        //time series array
    fftw_complex *fft_array;    //fft array
    fftw_plan plan;             //fft plane

    time_series = (double*) fftw_malloc( sizeof(double)*inz);
//working array in fftw_complex length of timeseries z
    fft_array = (fftw_complex*) fftw_malloc( sizeof(fftw_complex)*Nt);
//out array in fftw_complex length Nt
    plan = fftw_plan_dft_r2c_1d(inz, time_series, fft_array,
FFTW_EXHAUSTIVE);      //real to complex dft

    //Loop over pixels
    int x,y,z;
    for (y=0; y<iny; y++)
    {
        for (x=0; x<inx; x++)
        {

            //Do fft on pixel timeseries
            for (z=0; z<inz; z++)
            {
                //Build time array from in
                time_series[z] =
(double)(*(in+(iny*inx*z)+(inx*y)+x));
            }
            fftw_execute_dft_r2c(plan, time_series, fft_array);

            //copy to output array
            for (z=0; z<Nt; z++)
            {
```

```
                //store abs(F) into out array
                *(out+(iny*inx*z)+(inx*y)+x) = sqrt(
fft_array[z][0]*fft_array[z][0] + fft_array[z][1]*fft_array[z][1] );
            }
        }
    }
return 1;
}
```

Appendix B: Python module

This is the Python module (functions.py, a legacy name as it was originally a part of an FDM processing Python package) used to access the custom library functions and make them accessible.

```
"""
functions.py

Part of the fdm module.

Module containing functions for use on imaging with frequency division
multiplexing.
"""

import ctypes
import _ctypes
from sys import path
from os.path import join

import numpy as np
from scipy import ndimage

import iabFunctions.iab as iab
import DLLmanager as dllm


#---DLL stuff-----------------------------------------------------------
----------

# Paths to DLLs
# Drive may change as Python path list order is altered with installs
(lists have arbitrary ordering).
drive = path[2][:2]
# Rayleigh technique DLL
DLL_path = '{0:}\\IAB PhD\\dlls\\Rayleigh in
shearography\\'.format(drive)
DLL_demux = 'fast_rayleigh.dll' #old and certainly broken included as
it's old
DLL_fdm = 'fdm.dll' #new and working
# Unwrapping by LJMU method via DF coded DLL "dan_dll.dll"
DLL_dan_path = '{0:}\\IAB PhD\\dlls\\Dan DLL
(unwrapping)\\'.format(drive)
DLL_dan = 'dan_dll.dll'

# Remove the below print statement when module is installed on the lab
PC again.
print '<DLL files loading from external drive {0:}>'.format(drive)
```

```
# Create a DLL manager.
libs = dllm.dll_manager()


#-----------------------------------------------------------------------
----------

#-----------------------------------------------------------------------
----------

def set_filename(fn=None):
    """
    Sets timestamp for filenames and directory for raw data files.
    Used in loading data after capture.
    """
    if fn == None:
        fn = iab.qnamer()
    else:
        fn = fn
    return fn


def killer():
    """
    Breaks the program after logging abort. Used in capture loops to
break them
    and write into the log file that the capture was incomplete. Used
as a
    companion to iab.qlog for log writing in the loops.
    """
    fn = fn
    iab.qlog(fn, 'Capture aborted', fn[:7]+'log')
    raise Exception('Code execution aborted')


def fftfreq_fftw(n,d):
    """
    Use this to generate the correct array of frequencies for making
masks.

    This should be used instead of numpy.fft.fftfreq due to
diffferences in the
    fftw3 definitions.
    """
    N = (n//2) + 1
    f = np.zeros(n)

    #+ve freqs
    k = np.arange(0,N)
```

```
    f[ :N ] = (k/float(n*d))

    #-ve freqs
    if (n%2)==0:
        k = np.arange(-(N-2),0)
    else:
        k = np.arange(-(N-1),0)
    f[ N:] = (k/float(n*d))

    return f


#---------------------------------------------------------------------
----------

#---Masking-----------------------------------------------------------
----------


def create_mask(pass_freq, width, frequencies, dc_term=False,
harmonicsEven=False, harmonicsOdd=False):
    """
    DEPRECATED FUNCTION - Included for use by old code only.
    Use make_mask() instead.

    Call this to create a mask for the speckle_demux function.

    harmonicsEven, harmonicsOdd: Include windows for higher harmonics.
Allows
                two each of even and odd.
    dc_term: Allow DC term to pass if True, suppress DC if False
(recommended).
                Independant of window placement.

    Returns mask array with dtype == ctypes.c_int
        (correct for the speckle_demux functions)
    """
    print '<fdm.create_mask is deprecated. Use fdm.mask instead.>'
    modulation=pass_freq
    mask = np.zeros((frequencies.size), dtype=ctypes.c_int)
    #mask for fundamental
    mask[ (frequencies>(modulation-
width/2))*(frequencies<(modulation+width/2)) ] = True
    mask[ (frequencies<-(modulation-width/2))*(frequencies>-
(modulation+width/2)) ] = True
    #2nd harmonic
    mask[ (frequencies>((2*modulation)-
width/2))*(frequencies<((2*modulation)+width/2)) ] = harmonicsEven
    mask[ (frequencies<-((2*modulation)-width/2))*(frequencies>-
((2*modulation)+width/2)) ] = harmonicsEven
```

```
    #3rd harmonic
    mask[ (frequencies>((3*modulation)-
width/2))*(frequencies<((3*modulation)+width/2)) ] = harmonicsOdd
    mask[ (frequencies<-((3*modulation)-width/2))*(frequencies>-
((3*modulation)+width/2)) ] = harmonicsOdd
    #4th harmonic
    mask[ (frequencies>((4*modulation)-
width/2))*(frequencies<((4*modulation)+width/2)) ] = harmonicsEven
    mask[ (frequencies<-((4*modulation)-width/2))*(frequencies>-
((4*modulation)+width/2)) ] = harmonicsEven
    #5th harmonic
    mask[ (frequencies>((5*modulation)-
width/2))*(frequencies<((5*modulation)+width/2)) ] = harmonicsOdd
    mask[ (frequencies<-((5*modulation)-width/2))*(frequencies>-
((5*modulation)+width/2)) ] = harmonicsOdd
    #mask for dc
    mask[0] = dc_term
    return mask

def _rect_win(mask, freqs, centre, width):
    """
    Make rectangular window for mask functions
    """
    mask[ (freqs>(centre-width/2.0))*(freqs<(centre+width/2.0)) ] =
1.0
    mask[ (freqs<-(centre-width/2.0))*(freqs>-(centre+width/2.0)) ] =
1.0
    return mask


def make_mask(pass_freq, width, freqs, dc_term=False,
harmonicsEven=False, harmonicsOdd=False):
    """
    Call this to make a mask for the speckle_demux function.

    harmonicsEven, harmonicsOdd: Include windows for higher harmonics.
Allows
                two each of even and odd.
    dc_term: If True will be included in windows that span the DC
(F[0]), if
                False will be forced to zero in any mask spanning DC.

    Returns mask array with dtype == ctypes.c_int
        (correct for the speckle_demux functions)

    This function replaces create_mask().
    """
    print "<fdm.make_mask is deprecated. Use fdm.mask instead.>"
    mask = np.zeros(freqs.size, dtype=ctypes.c_double)
    # Mask for fundamental (always wanted).
```

```
    _rect_win(mask, freqs, pass_freq, width)
    # Even harmonics.
    if harmonicsEven == True:
        c = 2 * pass_freq
        while c <= freqs.max():
            _rect_win(mask, freqs, c, width)
            c = c + (2*pass_freq)
    # Odd harmonics.
    if harmonicsOdd == True:
        c = 3 * pass_freq
        while c <= freqs.max():
            _rect_win(mask, freqs, c, width)
            c = c + (2*pass_freq)
    #mask for dc term
    if dc_term == False:
        mask[0] = 0
    return mask

def make_mask_short(pass_freq, width, freqs, dc_term=False,
harmonicsEven=False, harmonicsOdd=False):
    """
    This function was used to make a mask for the fdm.speckle_demux.

    - Use fdm.mask in preference to this function.

    Now used in fdm.mask as the first call.

    Behaviour is eratic at low resolution (small number of images in
the
    time-series) as could produce masks with no windows.

    harmonicsEven, harmonicsOdd: Include windows for higher harmonics.
Allows
                two each of even and odd.
    dc_term: If True will be included in windows that span the DC
(F[0]), if
                False will be forced to zero in any mask spanning DC.

    Returns mask array with dtype == ctypes.c_int
        (correct for the speckle_demux functions)

    This function replaces create_mask().
    """
    if type(pass_freq) != type((0,0)): #if not a tuple, make it one
then loop over tuple.
        fs = (pass_freq,)
    else:
        fs = pass_freq

    masks = np.array(())
```

```
    # Do for positive frequencies only
    frequencies = freqs[freqs>=0]
    length = frequencies.size

    for f in fs:
        mask = np.zeros(length, dtype=ctypes.c_double)
        # Mask for fundamental (always wanted).
        _rect_win(mask, frequencies, f, width)

        # Even harmonics.
        if harmonicsEven == True:
            c = 2 * f # first even harmonic
            while c <= frequencies.max():
                _rect_win(mask, frequencies, c, width)
                c = c + (2*f) # next odd at c
        # Odd harmonics.
        if harmonicsOdd == True:
            c = 3 * f # first odd harmonic
            while c <= frequencies.max():
                _rect_win(mask, frequencies, c, width)
                c = c + (2*f) # next odd at c
        #mask for dc term
        if dc_term == False:
            mask[0] = 0

        masks = np.append(masks,mask)

    masks = masks.reshape((masks.size/length, length))
    masks = np.ascontiguousarray(masks)
    return masks

def mask(pass_freq, width, freqs, dc_term=False, harmonicsEven=False,
harmonicsOdd=False):
    """
    Call this to make a mask for the fdm.demux function.

    This is the de facto mask making function, replacing all others in
module.

    harmonicsEven, harmonicsOdd: Include windows for higher harmonics.
Allows
                two each of even and odd.
    dc_term: If True will be included in windows that span the DC
(F[0]), if
                False will be forced to zero in any mask spanning DC.

    Returns mask array with dtype == ctypes.c_int
        (correct for the speckle_demux functions)
    """
```

```
    masks = make_mask_short(pass_freq, width, freqs, dc_term,
harmonicsEven, harmonicsOdd)
    # Test all masks are made.
    for m in np.arange(masks.shape[0]):
        if masks[m].sum() == 0:
            pass_freq = pass_freq if
type(pass_freq)==__builtins__['tuple'] else (pass_freq,)
            alt_width = freqs[1] # guarantees a whole frequency
interval.
            print 'Mask width for {} Hz window insufficient; auto-
changed to {} Hz.'.format(pass_freq[m], alt_width)
            masks[m] = make_mask_short(pass_freq[m], alt_width, freqs,
dc_term, harmonicsEven, harmonicsOdd)
    masks = np.ascontiguousarray(masks)
    return masks


#-----------------------------------------------------------------
----------

#---Demux functions-----------------------------------------------
----------


def speckle_demux(images, freq_masks, speckle_patterns):
    """
    Wrapper for fast_rayleigh.dll
    images: image stack from camera (int8) as images[frame,y,x]
    freq_masks: frequency masks for each modulation frequency applied
(int8) as
                masks[mask, length]
    speckle_patterns: numpy array (numpy float) for speckle images to
be placed
                      as images[frame,y,x]
    """
#    DLL = ctypes.cdll.LoadLibrary(DLL_path+DLL_demux)
    DLL = libs.open_dll(DLL_demux, DLL_path)
    imagesPtr = images.ctypes.data
    shape = np.zeros(3)
    k = 0
    for s in images.shape:
        shape[k] = s
        k = k + 1
    freq_masksPtr = freq_masks.ctypes.data
    #no_masks = freq_masks.size / freq_masks.shape[0]
    no_masks = freq_masks.shape[0]
    outputPtr = speckle_patterns.ctypes.data

    res = DLL.speckle_demux_r2c( imagesPtr,
            ctypes.c_int(shape[0].astype(np.int)),
```

```
            ctypes.c_int(shape[1].astype(np.int)),
            ctypes.c_int(shape[2].astype(np.int)),
            ctypes.c_int(no_masks),
            freq_masksPtr,
            outputPtr )
#    _ctypes.FreeLibrary(DLL._handle)
    if res!=1:
        raise exception('Did not return success. Returned:
{0:}'.format(res))


def demux(images, freq_masks, speckle_patterns, method='linear',
debug=False):
    """
    demux

    Wrapper for fdm.dll

    Args must be contiguous data.

    images: Image stack from camera dtype == ctypes.c_uint8 as
images[frame,y,x]
    freq_masks: Frequency masks for each modulation frequency applied
- use
                fdm.masks(). dtype == ctypes.float as masks[mask,
length]
    speckle_patterns: Numpy array (numpy.zeros) for speckle images to
be placed
                as images[frame,y,x]
    method: Scaling to use on returned images
                - 'linear' = scaled to camera counts (linear response)
                - 'non-lin' = unscaled, non-normalised (non-linear
response)
                - 'complex' = Use a complex to complex transform
(untested)
    debug: Prints debug data before attempting to utilise the library.
    """
    # Check data is all contiguous. The DLL assumes contiguous arrays;
halt if not.
    if images.flags.c_contiguous != True:
        raise exception('Argument \"images\" is not contiguous in
fdm.demux().')
    if freq_masks.flags.c_contiguous != True:
        raise exception('Argument \"freq_masks\" is not contiguous in
fdm.demux().')
    if speckle_patterns.flags.c_contiguous != True:
        raise exception('Argument \"speckle_patterns\" is not
contiguous in fdm.demux().')
#    if _dimensionTest(images, speckle_patterns):
#        raise Exception('Image array dimensions do not match.')
```

```
    # Load the library.
#    DLL = ctypes.cdll.LoadLibrary(DLL_path+'fft_demux3.dll')
    DLL = libs.open_dll(DLL_fdm, DLL_path)

    # Pointers to contiguous data arrays.
    imagesPtr = images.ctypes.data
    freq_masksPtr = freq_masks.ctypes.data
    outputPtr = speckle_patterns.ctypes.data

    if debug == True:
            print imagesPtr, ctypes.c_int(images.shape[0]),
ctypes.c_int(images.shape[1]), ctypes.c_int(images.shape[2]),
ctypes.c_int(freq_masks.shape[0]), freq_masksPtr, outputPtr

    if method == 'linear':
        demuxer = DLL.demux
    elif method == 'non-lin':
        demuxer = DLL.demux_lite
    elif method == 'complex':
        demuxer = DLL.demux_complex
    else:
        raise exception('<Bad method in fdm.demux()>')

    # Get array dimensions. Allows for lower dimensional arrays by
using defaults.
    # Assumes there is always an array in time (zeroth  index
dimension).
    shape0 = images.shape[0]
    try:
        shape1 = images.shape[1]
    except:
        shape1 = 1
    try:
        shape2 = images.shape[2]
    except:
        shape2 = 1

    # Do demux via library.
    res = demuxer(
            imagesPtr,
            ctypes.c_int(shape0),
            ctypes.c_int(shape1),
            ctypes.c_int(shape2),
            ctypes.c_int(freq_masks.shape[0]),
            freq_masksPtr,
            outputPtr)

## A copy of the original call just in case.
#    # Do demux via library.
```

```
#    res = demuxer(
#            imagesPtr,
#            ctypes.c_int(images.shape[0]),
#            ctypes.c_int(images.shape[1]),
#            ctypes.c_int(images.shape[2]),
#            ctypes.c_int(freq_masks.shape[0]),
#            freq_masksPtr,
#            outputPtr)

    # Clear reference to library and do error check.
#    _ctypes.FreeLibrary(DLL._handle)
    if res!=1:
        raise exception('Did not return success. Returned:
{0:}'.format(res))


def speckle_demux_tohc(images, masks=[], img_out=None):
    """
    TOHC's Wrapper for demux.dll

    images      -   Image stack from camera (uint8) with shape =
(t,y,x)
    masks       -   Tuple/list of frequencies mask arrays (float64)
shape=(t)
    img_out     -   Optional output image array (float64) with shape
(n,y,x) whe
                    If None a new array is created.
    """
#    DLL = ctypes.cdll.LoadLibrary(DLL_path+'demux.dll')
    DLL = libs.open_dll('demux.dll', DLL_path)

    #pointer to images array and number of frames, rows, cols
    imagesPtr   = images.ctypes.data
    #stack the frequency masks together
    fmasks = np.ascontiguousarray( np.vstack( masks), dtype=np.float64
)
    if fmasks.shape[1] != images.shape[0]:
        raise Exception('Frequency masks wrong shape!')
    #dims input to dll
    dims_type = ctypes.c_int*4
    dims = dims_type()
    dims[0] = images.shape[0]
    dims[1] = images.shape[1]
    dims[2] = images.shape[2]
    dims[3] = fmasks.shape[0]
    #pointer to mask array
    maskPtr = fmasks.ctypes.data
    #make/check output array
    if img_out is None:
```

```
        img_out  = np.zeros( (fmasks.shape[0],)+images.shape[1:],
dtype=np.float64)
        made_out = True
    else:
        made_out = False
        if img_out.shape != images.shape[1:]:
            raise Exception('img_out array should have shape: ' +
                            str( images.shape[1:]))
        if img_out.dtype !=np.float64:
            raise Exception('img_out array should have dtype float64')
    #pointer to img_out
    outputPtr = img_out.ctypes.data

    res = DLL.demux_nc( imagesPtr, dims, maskPtr, outputPtr )

    # Clear reference to library, do error check, and return if
needed.
#    _ctypes.FreeLibrary(DLL._handle)
    if res!=1:
        raise Exception('DLL call failed, returned: '+str(res))
    if made_out==True:
        return img_out


def _dimensionTest(ts, oa):
    """
    will return True if arrays have a dimensional mismatch.
    ts: The time-series; oa: The output array for demuxed images.
    """
    #if ts.shape[1:] != oa.shape[1:]: ## does not work, but is
original version... kept just in case.
    if len(oa.shape) == 1:
        if ts.shape[1:] != (1,1):
            return True
        else: return
    else:
        if ts.shape[1:] != oa.shape:
            print '{} != {} (ts.shape[1:] !=
oa.shape)'.format(ts.shape[1:], oa.shape)
            return True
        else: return


#-------------------------------------------------------------------
----------

#---Visualisation of the FFTs---------------------------------------
----------
```

```
# Compute the power spectrum of each timeseries in an image bank

def power_spectrum(images, pow_out=None):
    """
    Calculate the power spectra of each pixel timeseries in the image
bank,
    images. Only the positive frequencies are calculated.
    (Wrapper for demux.dll)

    images      -   Image stack from camera (uint8) with shape =
(t,y,x)
    pow_out     -   optional output array ((t/2+1), y,x)
    """
    if images.dtype!=np.uint8:
        raise Exception('images array should have dtype uint8')

    DLL = libs.open_dll(DLL_fdm, DLL_path)

    #pointer to images array and number of frames, rows, cols
    imagesPtr   = images.ctypes.data

    N = (images.shape[0]//2)+1

    #make/check output array
    if pow_out is None:
        pow_out  = np.ones( (N,)+images.shape[1:],
dtype=ctypes.c_double)
    else:
        if pow_out.shape != (N,)+images.shape[1:]:
            raise Exception('pow_out array should have shape: ' +
                            str( (N,)+images.shape[1:]))
        if pow_out.dtype != np.float64:
            raise Exception('pow_out array should have dtype float64')

    #pointer to pow_out
    outputPtr = pow_out.ctypes.data

    # Get array dimensions. Allows for lower dimensional arrays by
using defaults.
    # Assumes there is always an array in time (zeroth  index
dimension).
    shape0 = images.shape[0]
    try:
        shape1 = images.shape[1]
    except:
        shape1 = 1
    try:
        shape2 = images.shape[2]
    except:
        shape2 = 1
```

```
    res = DLL.power_spectrum(
            imagesPtr,
            shape0,
            shape1,
            shape2,
            outputPtr )
    if res!=1:
        raise Exception('DLL call failed, returned: '+str(res))

    return pow_out


# Compute the fft of each timeseries in an image bank

def fft(images, fft_out=None):
    """
    Calculate the fft of each pixels timeseries in the image bank,
images
    Only positive frequencies are shown.
    (Wrapper for demux.dll)

    images      -   Image stack from camera (uint8) with shape =
(t,y,x)
    fft_out     -   optional output array ((t/2+1), y,x) of complex128
    """
    if images.dtype!=np.uint8:
        raise Exception('images array should have dtype uint8')

    #pointer to images array and number of frames, rows, cols
    imagesPtr   = images.ctypes.data

    N = (images.shape[0]//2)+1

    #make/check output array
    if fft_out is None:
        fft_out  = np.ones( (N,)+images.shape[1:],
dtype=np.complex128)
    else:
        if fft_out.shape != (N,)+images.shape[1:]:
            raise Exception('fft_out array should have shape: ' +
                            str( (N,)+images.shape[1:]))
        if fft_out.dtype !=np.complex128:
            raise Exception('fft_out array should have dtype
complex128')

    #pointer to pow_out
    outputPtr = fft_out.ctypes.data

    res = DLL.fft(
```

```
            imagesPtr,
            images.shape[0],
            images.shape[1],
            images.shape[2],
            outputPtr )
    if res!=1:
        raise Exception('DLL call failed, returned: '+str(res))

    return fft_out



#---------------------------------------------------------------------
----------

#---Phase-------------------------------------------------------------
----------


def phase5step(speckle):
    """
    Performs 5-step phase algorithm
    speckle should be 2d array shaped [phase_step_speckle, y, x]
    """
    phase = np.arctan2((2*(speckle[1] - speckle[3])), ((2*speckle[2])
- speckle[4] - speckle[0]))
    return phase

def scfilter(image, iterations, kernel):
    """
    Sine-cosine filter.
    kernel can be tuple or single value.
    Returns filtered image.
    """
    for n in range(iterations):
        image = np.arctan2(
            ndimage.filters.uniform_filter(np.sin(image),
size=kernel),
            ndimage.filters.uniform_filter(np.cos(image), size=kernel)
)
    return image

def unwrap_LJMU1(dp):
    """
    Unwrap the phase images using the Liverpool John Moores code
version 1 (THC code)
    """
#    dan_dll = ctypes.cdll.LoadLibrary(dan_dll_path)
    dan_dll = libs.open_dll(DLL_dan, DLL_dan_path)
```

```
    #int UnwrapLJMU1(float WrappedImage[], float OutImage[], int nx,
int ny)
    wrapped   = dp.astype(np.float32)
    unwrapped = np.zeros( dp.shape, dtype=np.float32)
    dan_dll.UnwrapLJMU1(wrapped.ctypes, unwrapped.ctypes, dp.shape[1],
dp.shape[0])
#    _ctypes.FreeLibrary(dan_dll._handle)
    return unwrapped

def unwrap_LJMU2(dp):
    """
    Unwrap the phase images using the Liverpool John Moores code
version 2
    and subtract a reference position. (THC code)
    """
#    dan_dll = ctypes.cdll.LoadLibrary(dan_dll_path)
    dan_dll = libs.open_dll(DLL_dan, DLL_dan_path)
    #int UnwrapLJMU1(float WrappedImage[], float OutImage[], int nx,
int ny)
    wrapped   = dp.astype(np.float32)
    unwrapped = np.zeros( dp.shape, dtype=np.float32)
    dan_dll.UnwrapLJMU2(wrapped.ctypes, unwrapped.ctypes, dp.shape[1],
dp.shape[0])
#    _ctypes.FreeLibrary(dan_dll._handle)
    return unwrapped



#-------------------------------------------------------------------
----------

#---Plotting stuff-----------------------------------------------------
----------


def scaling_limits(arrays_tuple, mask=None):
    """
    For an tuple of arrays (eg images) function returns the highest
and
    lowest values present extended to nearest power 10.

    mask: binary b&w image, black areas cover low quality, white for
valid data
        (approximate masking will be fine)

    Returns tuple of (min, max).
    """
    all_max = np.array(())
    all_min = np.array(())
    if mask == None:
```

```
        for array in arrays_tuple:
            array_max = array.max()
            array_min = array.min()
            all_max = np.append(all_max, array_max)
            all_min = np.append(all_min, array_min)
    else:
        for array in arrays_tuple:
            array_max = (array*mask).max()
            array_min = (array*mask).min()
            all_max = np.append(all_max, array_max)
            all_min = np.append(all_min, array_min)
    max = all_max.max()
    min = all_min.min()
#    min, max = _round(min,max)
    return min, max

def scale_point(phase, y, x):
    """ rescxale phase to a reference point"""
    phase = phase - phase[y,x]
    return phase

def rescale_zero(array, mask=None):
    """
    Rescale an array around the range limits of data defined by the
quality mask,
    re-centred on zero. If mask = None then array is centred on zero
based on
    values present in full array.
    """
    limits = scaling_limits(array, mask)
    scaled_array = (array + limits[0]) - ((limits[1] - limits[0]) /
2.0)
    return scaled_array

# This one works like a charm! :)
def rescale_zero2(array, mask=None):
    """
    Rescale an array around the range limits of data defined by the
quality mask,
    re-centred on zero. If mask = None then array is centred on zero
based on
    values present in full array.
    """
    if mask == None:
        scaled_array = array - array.mean()
    else:
        scaled_array = array - (np.sum(array * mask) / array.size)
    return scaled_array

def _round(min, max):
```

```
    """
    Rounds to nearest power of 10 floor and ceiling respectively of
min and max.
    Gives plot scaling limits that lie outside the values contained in
the plot.
    """
    if max != 0:
        max_sign = max/np.abs(max)
        max = np.abs(max)
        max = max_sign * (np.floor(np.log10(max)) * 10) *
np.ceil(max/(np.floor(np.log10(max))*10))
    if min != 0:
        min_sign = min/np.abs(min)
        min = np.abs(min)
        min = min_sign * (np.floor(np.log10(min)) * 10) *
np.floor(min/(np.floor(np.log10(min))*10))
    return min, max

def zero(phasemap, zero_pixel_index):
    """
    Zero the phase to the value of the zero pixel, given as a tuple of
a (y,x)
    index.
    """
    y, x = zero_pixel_index
    out = phasemap - phasemap[y, x]
    return out



#def speckle_demux_fdm(images, freq_masks, speckle_patterns):
#    """
#    Wrapper for fast_rayleigh.dll
#    images: image stack from camera (int8) as images[frame,y,x]
#    freq_masks: frequency masks for each modulation frequency applied
(int8) as
#                masks[mask, length]
#    speckle_patterns: numpy array (numpy complex) for speckle images
to be
#                placed as complex images[frame,y,x]
#    """
#    if type(speckle_patterns) != type(numpy.complex()):
#        speckle_patterns = numpy.complex(speckle_patterns)
#    DLL = ctypes.cdll.LoadLibrary(DLL_path)
#    imagesPtr = images.ctypes.data
#    shape = images.shape
#    freq_masksPtr = freq_masks.ctypes.data
#    no_masks = freq_masks.shape[0]
#    outputPtr = speckle_patterns.ctypes.data
#
```

```
#    res = DLL.speckle_demux_r2c( imagesPtr,
#            ctypes.c_int(shape[0]),
#            ctypes.c_int(shape[1]),
#            ctypes.c_int(shape[2]),
#            ctypes.c_int(no_masks),
#            freq_masksPtr,
#            outputPtr )
#    _ctypes.FreeLibrary(DLL._handle)
#    if res!=1:
#        raise exception('Did not return success. Returned:
{0:}'.format(res))
```